VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

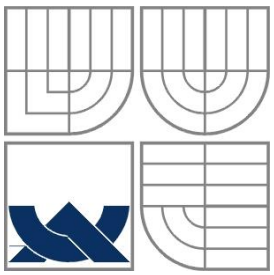
SKRYTÍ DAT V POČÍTAČOVÝCH SÍTÍCH

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

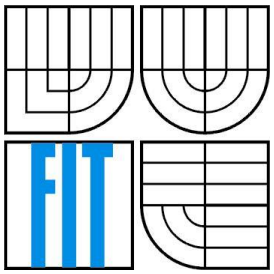
AUTOR PRÁCE
AUTHOR

Bc. MARTIN HREBÍČEK

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SKRYTÍ DAT V POČÍTAČOVÝCH SÍTÍCH

HIDING DATA IN COMPUTER NETWORKS

DIPLOMOVÁ PRÁCE
MASTER‘S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MARTIN HREBÍČEK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. LIBOR POLČÁK

BRNO 2013

Zadání diplomové práce/13202/2012/xhrebi01

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2012/2013

Zadání diplomové práce

Řešitel: **Hrebíček Martin, Bc.**
Obor: Počítačové sítě a komunikace
Téma: **Skrytí dat v počítačových sítích**
Hiding Data in Computer Networks
Kategorie: Počítačové sítě

Pokyny:

1. Seznamte se s principy fungování systému pro zákonné odposlechy v počítačových sítích.
2. Analyzujte a popište možnosti oklamání systému pro zákonné odposlechy.
3. Navrhněte způsob ukrytí dat před systémem pro zákonné odposlechy.
4. Navrhněte softwarový nástroj umožňující ukrýt data před systémem pro zákonné odposlechy.
5. Návrh implementujte.
6. Analyzujte možnosti detekce implementovaného útoku a vytvořte nástroj pro odhalení ukrytých dat.
7. Zhodnoťte vytvořené nástroje.

Literatura:

- Bellowin. Wiretapping the net. 2000. The Bridge 20 (2) p. 21-26.
- Cronin, Sherr and Blaze: On the Reliability of Network Eavesdropping Tools. IFIP Advances in Information and Communication Technology, 2006, Volume 222/2006, 199-213.
- A další podle pokynů vedoucího.

Při obhajobě semestrální části diplomového projektu je požadováno:

- Body 1 až 4.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci ročníkového a semestrálního projektu (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Polčák Libor, Ing.**, UIFS FIT VUT
Datum zadání: 17. září 2012
Datum odevzdání: 22. května 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Tato diplomová práce se zaobírá skrytím dat v internetovém provozu. Obsahuje popis systému pro zákonné odposlechy. Následně jsou uvedené různé možnosti skrytí dat. Praktická část práce se zabývá přenosem dat protokolů HTTP a HTTPS v rámci falešného VoIP hovoru. Vytvořený balík programového vybavení se skládá se dvou spolupracujících částí: klientské a serverové. Data přenášená mezi klientskou a serverovou částí mají formát multimediálních dat VoIP hovoru. Pokud uživatel nebo internetový server nemá žádné data k odeslání, tak se mezi klientskou a serverovou částí přenášejí náhodná data simulující VoIP hovor. Práce následně popisuje způsob detekce skrytých dat ve VoIP hovoru.

Abstract

This diploma thesis deals with hiding data in the Internet traffic. It contains a description of the law interception. Various possibilities of hiding data are mentioned. The practical part of this thesis consists of an application that hides the data of HTTP and HTTPS protocols in a fake VoIP call. The application consists of two parts: a client and a server. Data transmitted between the client and the server parts are masked as multimedia data of the VoIP call. When a user or Internet server does not transmit any data, random data are transmitted between client and server parts in order to simulate the VoIP call. Then, the thesis focuses on detection of the attack.

Klíčová slova

Zákonní odposlech, skrytí dat, skryté kanály, VoIP, steganografia

Keywords

Lawful interception, hiding data, covert channels, VoIP, steganography

Citace

Martin Hrebíček: Skrytí dat v počítačových sítích, diplomová práce, Brno, FIT VUT v Brně, 2013

Skrytí dat v počítačových sítích

Prehlásenie

Prehlasujem, že som túto diplomovú prácu vypracoval sám pod vedením Ing. Libora Polčáka.

……
Martin Hrebíček
14. mája 2013

Poďakovanie

Ďakujem svojmu vedúcemu Ing. Liborovi Polčákovi za rady a odbornú pomoc pri vypracovaní diplomovej práce.

Obsah

1 Úvod

Komunikácia prostredníctvom internetu je dnes veľmi rozšírená. Dôležitú úlohu v nej zohráva bezpečnosť a ochrana súkromia. Z rôznych dôvodov sa snažia niektorí ľudia vzájomnú výmenu informácií utajiť. Nie vždy je to ľahko realizovateľné. Ako prvý zdanlivo dostatočný spôsob sa môže zdať šifrovanie. Poskytuje utajenie obsahu komunikácie. Ale v určitých podmienkach to nie je dostatočné riešenie. Pretože aj samotný fakt, že nejaká výmena informácií nastala, môže vzbudiť nechcenú pozornosť.

Spôsob ako skryť dáta môže spočívať v ich vložení alebo zamaskovaní do iných dátových prenosov, ktoré sú bežne používané. Takto vložené dáta zostanú skryté pred ostatnými užívateľmi aj bez použitia šifrovania. Dôležité je utajiť spôsob, ktorým boli dáta do prenosu vložené.

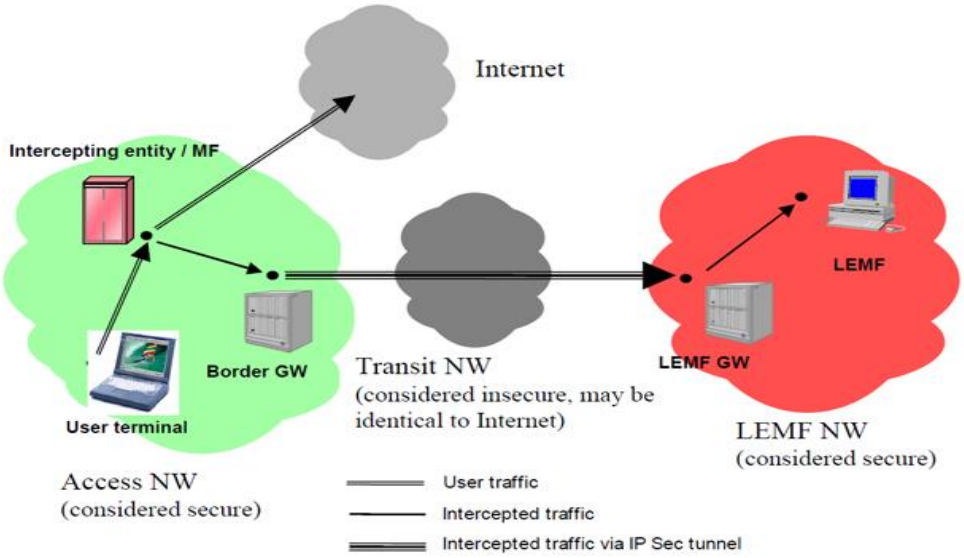
Ako vhodný príklad je možné uviesť snahu o komunikáciu podliehajúcu cenzúre. Pomocou Tor siete [1] je možné relatívne tajne komunikovať. Jeho autori sa snažia aby sa dáta prenášané cez tento protokol tvárili ako obyčajný prenos šifrovaných dát (napríklad bežné prehliadanie internetu cez protokol HTTPS). Tak ako všetko iné, aj toto sa dá zneužiť. Utajené komunikácie zakrývajúce aj zdroj aj cieľ sú veľmi výhodné pre nelegálnu činnosť. Môže sa jednať o vyzrádzanie určitých informácii (prekonanie bezpečnostných politík) alebo to môže byť snaha o utajenie komunikácie medzi C&C (command-and-control) servermi a počítačmi v botnete. Preto je dôležité o týchto spôsoboch vedieť aby sa mohlo zabrániť ich zneužitiu. Predovšetkým ide o ich detekciu ale aj o tvorbu aplikácii a protokolov, ktoré by boli voči tomu čo najviac odolné.

Táto práca sa zameriava na popis a implementáciu skrytia dát vo falošnom VoIP hovore. Zdrojové dáta predstavuje komunikácia protokolmi HTTP a HTTPS. Tieto dáta sú následne transformované do podoby RTP paketov, aby napodobňovali VoIP hovor. Tiež bude ukázané ako je možné rozpoznať či sa jedná o skutočný VoIP hovor alebo o nejaké dáta, ktoré sa maskujú ako VoIP hovor.

Práca sa skladá z ôsmych kapitol. V druhej kapitole sa nachádza popis zákonných odposluchov. Kapitola obsahuje spôsob ako sa odpočúvané dáta získavajú a vzťah odposluchov k tejto práci. Nasleduje tretia kapitola, ktorá obsahuje popis techník skrytia dát v počítačových sieťach. V štvrtej kapitole sa nachádza popis konkrétneho spôsobu skrytia dát v počítačových sieťach a jeho vlastnosti. Bude sa jednať o skrytie dát protokolov HTTP a HTTPS do falošného VoIP hovoru. Nasleduje kapitola 5, ktorá obsahuje návrh aplikácie proxyvoip. Šiesta kapitola popisuje implementáciu aplikácie proxyvoip. V kapitole 7 sú ukázané spôsoby ako je možné zistiť, že nejaké dáta sa maskujú ako VoIP hovor. Na základe týchto spôsobov je navrhnutá aplikácia zistivoip. V kapitole 8 sú uvedené dôležité vlastnosti aplikácií proxyvoip a zistivoip. Kapitola popisuje klady a zápory aplikácii a uvádza možné zlepšenia do budúcnosti. Obsah práce je zhrnutý v závere.

2 Systémy pre zákonné odposluchy

Zákonné odposluchy sú štandardizované v normách úradu ETSI (European Telecommunications Standards Institute) [2, 3]. Informácie získavané odposluchom sa delia na dva druhy. Sú to metainformácie (Intercept Related Information - IRI) popisujúce komunikáciu odpočúvaného. Prípadne je to záznam samotnej komunikácie (Content of Communication - CC).



Obrázok 1 Ilustrácia systému zákonného odposluchu. Prevzatý z ETSI [3].

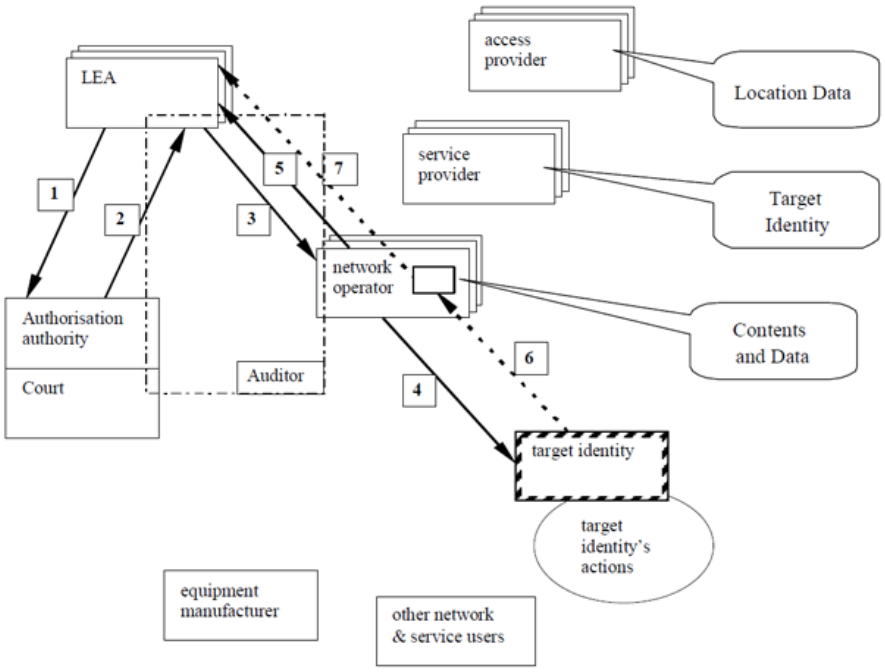
Odposluch (na obrázku 1) nesmie obmedzovať ani funkčnosť ani výkonnosť komunikačných prostriedkov, z ktorých informácie získava. Taktiež musí byť izolovaný od odpočúvaného objektu. Najčastejšie bude systém pre zákonné odposluchy (Lawful Interception System - LIS) použitý na strane poskytovateľa počítačových sietí (Network operator alebo Access provider – NwO, AP) alebo poskytovateľa služieb (Service provider - SvP). LEA (Law Enforsment Agency – orgány činné v trestnom konaní) prijíma dáta na opačnej strane. Na komunikáciu používa tri HI (Handover Interface) rozhrania.

- **Rozhranie H1** slúži na administráciu. Pomocou neho sa zadávajú a odoberajú odposluchy. Možné problémy sú tiež oznamované na toto rozhranie. Na strane LEA môže prebiehať preklad medzi internými protokolmi a protokolmi používanými na strane dodávajúcej odposluchové dáta.
- **Rozhranie H2** slúži na prenos správ IRI. Správy môžu byť prijímané z rôznych zdrojov. Závisí, ktoré sú najvhodnejšie na daný odposluch (napríklad prístupový uzol, prepínací uzol).
- **Rozhranie H3** sa používa pre prenos správ CC. Dáta sú prijímané podľa typu odpočúvanej identity. Ak sa odpočúva emailová komunikácia, tak sa príjmu emaily. V prípade odposluchu na úrovni ATM kanálu, sa na rozhranie posiela ATM data stream.

Nesmie sa stať, že odposluch (príklad použitia sa nachádza na obrázku 2) sa aktívne zúčastní komunikácie (napríklad žiadosť o preposlanie dát). Spolu so zachytenými dátami je v niektorých prípadoch vhodné priložiť aj prostriedky na rozkódovanie samotnej komunikácie (môže sa jednať napríklad o VoIP). Závisí aj na tom, z ktorej vrstvy OSI modelu sú informácie získavané. V prípade, že sa jedná o nižšie vrstvy, môže byť posielaný na zberné zariadenie kompletný dátový prúd dát bit po bite. Tento postup je sa deje, ak sa presne nedá určiť o aký typ komunikácie sa jedná. Zo všetkých prenesených dát sa potom dá ľahšie zrekonštruovať pôvodná komunikácia. Obecne sa dá povedať, že z čím vyššej vrstvy sú dáta zachytené, tým ľahšie a presnejšie je možné komunikáciu zrekonštruovať. Je dôležité aby sa ukladali len informácie vzťahujúce sa k odpočúvanému objektu. V prípade, že sa na odpočúvanom médiu vyskytuje aj komunikácie iných objektov (napr. pri použití protokolu ATM), je nutné ich odfiltrovať.

V niektorých krajinách sa v tomto prípade odposluch neumožňuje. Zariadenie za uloženie zachytenie odpočutých informácii by malo byť pripojené k zdroju dát aspoň na takej istej rýchlosti ako sa očakáva prenos odpočúvanej komunikácie. V opačnom prípade by sa dáta nestíhali odosielať a ukladať. V prípade bufferovania dát na strane zdroja je možné dočasne uložiť len určité množstvo dát a prípadné ďalšie dáta sa musia zmazať respektíve neodoslať k zbernému zariadeniu. Taktiež dôležité je aj zabezpečenia spojenia medzi LIS a zberným zariadením. Tretie strana sa nesmie k odpočutým informáciám dostať.

Príklad použitia zákonného odposluchu [2]:



Obrázok 2 Príklad systému zákonného odposluchu. Prevzatý z ETSI [2].

1. LEA požiada o právo na odposluch (napríklad súd)

2. LEA právo na odposluch získa
3. LEA požiada NWO/AP/SvP o spoluprácu.
4. NWO/AP/SvP pripraví prostriedky na odposluch.
5. NWO/AP/SvP oznámi LEA, že odposluch je pripravený.
6. IRI a CC sú poskytnuté od odpočúvanej identity do NWO/AP/SvP
7. IRI a CC sú poskytnuté od NWO/AP/SvP do LEMF (Law Enforcements Monitoring Facility – zber odpočutých dát)
8. NWO/AP/SvP ukončujú odposluch po žiadosti z LEA alebo po skončení doby platnosti
9. NWO/AP/SvP oznámia koniec odposluchu LEA

Z hľadiska tejto diplomovej práce je dôležité uvedomiť si, z ktorého bodu komunikačnej infraštruktúry sa zachytené dáta poskytujú LEA. Získavajú sa na strane NWO/AP/SvP, ktoré poskytnú úplný dátový prúd (prípadne iba IRI) do LEMF. LEA má k dispozícii kompletný obraz sieťovej komunikácie. Jediný spôsob ako skryť dáta pred odposluchom, je zamaskovať ich v bežnom prenose.

3 Spôsoby skrytia dát

Kapitola popisuje teoretické možnosti skrytia dát. Následne, v podkapitole 3.1, sú uvedené príklady realizácie niektorých druhov skrytých kanálov. Podkapitola 3.2 popisuje skrytie dát pomocou zmätenia odposluchu.

Existuje niekoľko rôznych možností ako ukryť dáta pred nežiaducim odhalením. Jednou z možností je použitie *šifrovania*. Použitím vhodného šifrovacieho algoritmu a dostatočne dlhého kľúča zostanú dáta tajné. Tento spôsob však nezabezpečuje utajenie faktu, že nejaké tajné dáta vôbec existujú. Preto boli vymyslené metódy ako vložiť utajované dáta do iných. Jednou z týchto metód je *steganografia*. Jednou z možností aplikácie steganografie sú *skryté kanály*. Aj v tomto prípade je možné dáta zašifrovať kvôli vyššiemu stupňu bezpečnosti. Predovšetkým je dôležité, že odposluch nevie, že sa nejaké dáta prenášajú.

Iná možnosť je falzifikovanie odosielateľa a príjemcu za nejaké vierohodné entity. Avšak pretrváva problém, že je možné vidieť samotný prenos dát.

Pre použitie skrytých kanálov môžu viesť rôzne dôvody. Napríklad prenos dát, ktoré sú oficiálnymi cestami blokované. Môže sa jednať o obídenie Data Loss Prevention (DLP) softwaru. Skryté kanály môžu byť používané aj pri obchádzaní cenzúry internetu [4]. Rôzne cenzúrovacie algoritmy sa snažia obmedziť prenosu špecifických informácii do vnútra sieti alebo von. Pričom sa snažia blokovať komunikáciu, ktorá obsahuje nepovolené informácie. Toto sa môže diať pomocou sledovania obsahu paketov. Programy na cenzúrovanie informácie sa snažia nájsť určité slová. Na základe ich výskytu následne zamedziť komunikácii. Týmto spôsobom je filtrovaný web, email a aj IM (instant messaging) komunikácia. Ak je prenos dát pred týmto kontrolórom skrytý, nemôže v ňom hľadať dané slová a preto ho ani obmedziť. Práve v tomto prípade by samotné šifrovanie nestačilo. Pretože výskyt podozrivého zašifrovaného prenosu môže znamenať pokus obchádzanie kontroly a je automaticky zablokovaný.

Ďalšie použitie sa týka programov, ktorých prenos má byť nespozorovateľný. Môže sa jednať napríklad o spyware, ktoré prenáša údaje útočníkovi. Platí to pre trójske kone, útočník sa snaží zakryť komunikáciu s počítačom, aby znížil šancu na detekciu samotného programu.

Dôvody použitie skrytých kanálov môžu zahŕňať snahu o komunikáciu, ktorá nie je povolená. Napríklad: nespokojný zamestnanec sa snaží vyzradiť firemné tajomstvo. Pretože použitie USB kľúča alebo napaľovanie CD môže byť monitorované. A tiež prístup na internet a email podlieha DLP. Preto sa snaží ukryť dáta do iných a bežných dát, ktoré cez sieť prejdú bez všimnutia. Môže sa jednať o používanie IM vo firmách. Prípadne návšteva internetových stránok, ktoré nie sú povolené.

3.1 Skryté kanály

Na využívanie skrytia informácii počas prenosu je vhodné používať počítačovú sieť, ktorá dosahuje určitú kvalitu prenosu. Je vhodné aby sa prenášalo čo najviac informácií, do ktorých sa môžu ďalšie informácie skryť. S príchodom internetu sa výrazne zjednodušil spôsob prenosu informácii. Postupným sprístupňovaním internetu širšiemu okruhu užívateľov a aj jeho samotným zrýchľovaním vznikol priestor na použitie skrytých kanálov. Prenos veľkého počtu obyčajných (z pohľadu odposluchu nezávadných dát) dát je vhodným predpokladom použitia skrytých kanálov. Z dôvodu maskovania samotného prenosu skrytých dát je ich pomer voči normálnym dátam veľmi malý (vo väčšine prípadov). Naopak prípadný zvýšený výskyt skrytých kanálov v komunikácii môže uľahčiť ich detekciu. Taktiež je vhodné aby pripojenie k internetu malo stabilné vlastnosti. Rôzne výkyvy rýchlosti a prípadne kvality spojenia môže mať negatívne vplyvy na určité typy skrytých kanálov. Predovšetkým časové skrytie údajov bude mať problémy pri napríklad slabom signáli Wi-Fi, kde sa budú musieť opakovane prenášať dáta a ešte s rôznym časom prenosu.

Skryté kanály možno rozdeliť na dva hlavné druhy:

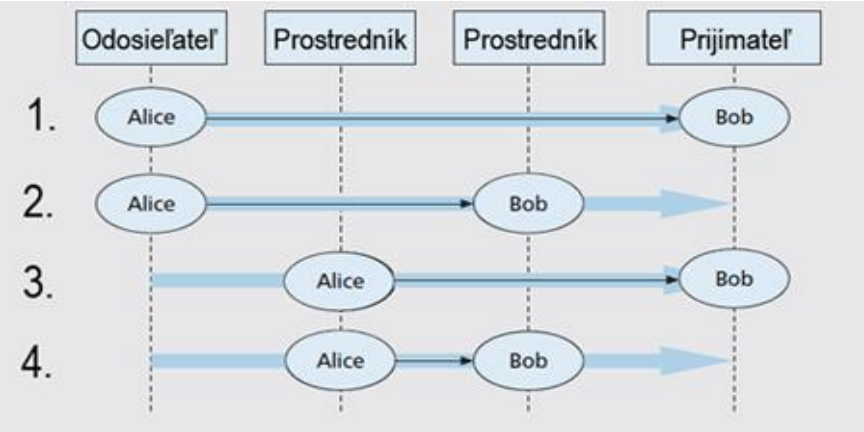
- Pamäťové skryté kanály – samotné dáta sú určitým spôsobom vložené do komunikácie
- Časové skryté kanály – informácia je prenášaná formou časových signálov, môže sa jednať o prenos normálnych dát v presne daných intervaloch

Ilustrácia skrytého kanálu je možná na väzenskom probléme. Väzenský problém bol navrhnutý Simmonsom [5] ako ukážka potreby kryptografie pri komunikácii. Ale Zander et al. [6] na tom ukazuje aj potrebu skrytých kanálov. Vo väzení sú dve osoby Alica a Bob. Aby si mohli dohodnúť plán úteku potrebujú spolu komunikovať. Ale strážnik Wendy odpočúva všetku komunikáciu. Preto sa snažia väzni dohovoriť tak, aby strážnik nič nespozoroval. Craver [7] popísal tri druhy strážnikov, podľa manipulácie so správami:

- Pasívny - iba pozoruje komunikáciu ale nemení ju.
- Aktívny - môže meniť komunikáciu ale sémanticky musí zostať rovnaká.
- Záškodnícky (Malicious) - môže meniť správu ľubovoľne.

Tento scenár môžeme použiť ako komunikáciu dvoch entít – počítačov Alice a Boba na sieti, ktorá môže byť monitorovaná (napríklad nejakým kontrolným softwarom).

Pri skrytých kanáloch je tiež dôležité, v ktorom bode komunikácie sa dáta do nej vkladajú a extrahujú (viď obrázok 3). S tým súvisí, kde sa nachádza miesto, z ktorého sa získavajú dáta na odposluch. Podľa toho je stúpa zložitosť vloženia resp. extrakcie dáta z prenosového kanálu [8].



Obrázok 3 Možné pozície odposluchu počas sieťovej komunikácie. Prevzaté od Zander et al. [6].

Prvé rozloženie (viď obrázok 3) je najvhodnejšie pre prenos utajovaných informácií. Pretože odosielateľ dátový prenos vytvára a prijímateľ ho prijíma a u neho aj končí. Ľahšie sa dáta vložia do niečoho, čo sa na to môže pripraviť a prispôsobiť. V nasledujúcich rozloženiach je to obtiažnejšie. V druhom rozložení sa musia dáta vyextrahovať a prenos pokračuje ďalej k jeho prijímateľovi. Tretie rozloženie môže byť problematické. Pretože samotný prenos už je kompletný a musia sa dáta do neho nejako vsunúť. Pri niektorých druhoch komunikácie to nemusí byť problém, ale pri iných to môže spôsobiť výraznejšie komplikácie. Napríklad je nutné vstupné dáta zmeniť a zmenšiť ich veľkosť bez straty sémantickej informácie. A až následne do ušetreného miesta vložiť utajované dáta.

Zander et al. [6] navrhuje rozdeliť druhy skrytých kanálov podľa mechanizmu skrytia dát a nie podľa toho, na ktorých vrstvách OSI modelu sa nachádzajú. Z pohľadu množstva prenesených dát majú skryté kanály rôzne vlastnosti. Niektoré majú malú priepustnosť a nie sú určené na prenos väčších súborov. Ale ak je požadované preniesť len malé množstvo informácii alebo je k dispozícii dostatok času, tak potom to nie je výraznejšia vada. Nasledovné typy skrytých kanálov sú uvedené podľa toho aby umožňovali preniesť väčšie množstvo informácii. Prípadne aby boli implementovatelné, bez potreby zachovania veľmi špecifických podmienok. Určité spôsoby skrytia dát dosahujú dobré výsledky v len v špecifických podmienkach (napríklad obmedzený počet užívateľov alebo použitých protokolov). Ale na internete, kde je veľa rôznych rušivých vplyvov, už nedosahujú také uspokojivé výsledky.

Pri ďalej uvedených spôsoboch a druhoch utajenia prenesených dát nie sú uvedené spôsoby využívajúce prvú a druhú vrstvu modelu OSI [9], pretože sa očakáva prenos informácii po sieti (internete). Vtedy sa nepožaduje, že by sa mohli nastaviť všetky aktívne prvky, tak aby bol prenos utajených informácií nedetekovateľný. Ďalej uvedené spôsoby sú platné pre vrstvy 3 a vyššie OSI modelu. To je prevažne softwarová záležitosť. Preto by nemal byť problém s manipuláciou dát a snahou o prenesenie takto upravených dát až k adresátovi.

Nevyužité bity v hlavičkách

Dáta v skrytých kanáloch sú zakódované pomocou bitov, ktoré nie sú využívané alebo sú rezervované v hlavičkách rámcov alebo paketov. Tento spôsob je možný nepotrebnosťou niektorých metadát (vždy alebo v niektorých špecifických prípadoch). Pri plánovaní nových protokolov sa považovalo za potrebné prenášať určité typy metadát. Prípadne sa nechávali niektoré polia voľné pre budúce rozšírenie. Niektoré údaje sa postupom času prestali používať alebo majú stále tú istú nemennú hodnotu. Tieto vlastnosti umožňujú fungovanie skrytých kanálov, ktoré sa spoliehajú na možnosť použitia nepotrebných metadát. Prípadne, zmeny polí hlavičky, ktoré sa používajú, ale nie sú pre prenos úplne dôležité. Tiež je možné, že prijímateľ vie o prípadnej zmene určitých dát podľa tajných informácii. Vie aké údaje tam majú byť pôvodne a tie potom môže zrekonštruovať. Tento spôsob prenosu demonštruje Handel [10], on používa IP hlavičku, pole Type of Service alebo Flags pole v protokole TCP.

Možné protiopatrenia pomôžu týmto typom skrytých kanálom zamedziť. Môže sa jednať o základné nastavovanie nepoužívaných hlavičiek na nulu, prípadne inú štandardnú hodnotu. Týmto sa eliminujú úplne základné typy skrytých kanálov, ktoré využívajú iba určité nepoužívané polia v hlavičkách. Na detekciu a elimináciu kanálov, ktoré upravujú iné normálne používané hodnoty v poliach hlavičky pomocou nejakého, pre daný protokol bežného rozloženia je to zložitejšie. Pre tento prípad sa väčšinou vytvára fingerprint protokolu a potom sa porovnáva či sa veľmi nevychyľuje od bežných zaznamenaných hodnôt.

Rozšírenie hlavičky a zarovnanie

Tajné informácie sa môžu skryť aj v rozšíreniach hlavičky niektorých protokolov (napríklad IPv6). Protokol IPv6 poskytuje priestor na pridanie hlavičiek podľa potreby aktuálneho prenosu. V tomto prípade sa dáta môžu priamo zakódovať do týchto doplnkových hlavičiek. Toto navrhuje Graf [11], predpokladá nastavenie option flagu, tak aby sa žiadne doplnkové hlavičky neočakávali a potom sa do nich môžu vložiť informácie.

Niektoré protokoly predpokladajú určitú minimálnu veľkosť paketu. Ak skutočné dáta ju nedosahujú, preto je možné ich doplniť skrytými dátami. Napríklad sa jedná o protokol Ethernet, ktorý pracuje s minimálne 64 bytmi [12].

Zamedziť týmto typom kanálov je možné zrušením doplnkových hlavičiek. Ak sa nejaká doplnková hlavičku vymyká bežného prenosu, odstráni sa z paketu. Týmto sa môže ohroziť funkcionalita bežného prenosu, ak sú doplnkové hlavičky používané na prenos štandardných metadát. Ale väčšinou sa nič nestane a iba sa zlepší zabezpečenie. Ak by sa zistilo, že nejaká aplikácia kvôli tomuto nefunguje ako má, po dôkladnejšom preskúmaní sa môže špecificky nejaká hodnota povoliť alebo sa vymyslí nejaký workaround. Proti paddingu sa môže postupovať povolením určitých veľkostí paketu [13]. Týmto sa zhorší kapacita prenosu.

Kontrolný súčet

Skrytie dát poskytuje aj možnosť upravenia CRC (Cyclic redundancy check). Dáta sa vložia do paketu a do hlavičky sa pripojí rozšírenie, aby kontrolný súčet bol správny. Avšak toto sa dá spraviť len pomocou rozšírenia hlavičiek a preto uplatnenie nie až tak veľké. Protokol UDP má CRC ako voliteľný parameter a preto Fisk et al. [14], navrhoval použiť túto vlastnosť ako 1 bit skrytého kanálu, podľa toho či CRC je alebo nie je prítomný v pakete.

Možná eliminácia toho typu kanálu môže byť obmedzenie použitia doplnkových hlavičiek, podobne ako bolo spomenuté v predošlom odseku. A tiež správne vypočítaný CRC pre každý paket.

Wi-Fi

Potvrdzovacie správy v bezdrôtových LAN sieťach (ACK - acknowledgement) je možno využiť ako skryté kanály. Môže sa jednať o ich počet, prípadne čas v akom sú poslané. Taktiež je možné použiť invalidné rámce (účelne nesprávne udaný CRC). Následne v poli cieľovej adresy je možné uviesť „magické číslo“ [15]. Počet opakujúcich sa správ, obecne sa môže jednať o hocijaké, navrhuje Krätzer et al. [16]. Komunikácia pozostáva zo zámerného posielania zdvojených rámcov a na druhej strane ich detekciou.

Protiopatrenie voči použitiu nesprávneho CRC je používanie iba jeho správnej hodnoty. Ale toto sa v bezdrôtových sieťach vynútiť nedá. Ale môže to vzbudiť pozornosť, že sa v sieti deje niečo neštandardné. Ďalej je možné nastaviť obmedzenie niektorého typu správ po presiahnutí určitého prahu. Niektoré iné typy sa nedajú, pretože to protokol povoľuje vo svojom štandarde. Napríklad preposielanie istých typov rámcov alebo ACK sa tiež nedá úplne obecne zamedziť alebo označiť za nesprávne. Pri používaní Wi-Fi môžu nastať rôzne podmienky a môže to byť prirodzené. A prípade eliminácie časových skrytých kanálov je možné použiť buffer, ktorý na nejaký čas pozdrží pakety a preto zamedzí časovým prenosom informácie.

Protokol HTTP (Hypertext Transfer Protocol)

V prospech protokolu HTTP hovorí jeho veľké rozšírenie. A taktiež veľká variabilita dát cez neho prenášaných. Jeho využitie na prenos utajených informácii môže byť pomocou obyčajných požiadaviek na stránku a príslušnej odpovedi na požiadavku. Jedná sa o HTTP Request na stránku pomocou dopredu dohodnutých parametrov, prípadne použitých polí v hlavičke. Server následne odpovie internetovou stránkou obsahujúcou obrázky. Tie majú v sebe zakódované informácie pomocou steganografie [17].

Prípadné zamedzenie tohto typu skrytých kanálov môže spočívať v použití zoznamov povolených cieľových adries. Vhodná je aj kontrola parametrov týchto HTTP Requestov a ich následných odpovedí.

Protokol SSH

Ukrytie dát spočíva v nahradení MAC hlavičky v každom pakete (Message Authentication Code) [18]. Odosielateľ ju nahradí zašifrovanými dátami. Druhá možnosť je použiť krátku MAC a zbytok doplniť skrytými dátami. Táto metóda je výhodnejšia. Môže sa použiť aj ak odosielateľ skrytej informácie a jej prijímateľ nie sú zdroj, respektíve cieľ samotnej SSH komunikácie.

Detekcia tohto typu prenosu utajených informácii je relatívne jednoduchá, pretože sa komunikácia dosť odchyľuje od bežných SSH spojení. Mení sa dĺžka paketu oproti bežným spojeniam, na toto upozorňuje Zander et al. [6].

Protokol FTP

Možnosť použitia skrytých kanálov spočíva v ich zakódovaní do FTP príkazov, pretože RFC ukazuje použitia kódovania v ASCII znakoch. Zou et al. [19] ukazuje algoritmus vytvorenia príkazu, tak aby obsahoval skrytú informáciu. Iný spôsob poukazuje na zasielanie správ FTP NOOP počas neaktívnej doby komunikácie. Protiopatrenie k tomuto skrytému kanálu môže byť aplikovanie entropie. Zaznamenajú sa bežné vlastnosti normálnej komunikácie užívateľov s FTP serverom. A následne sa zisťuje ako veľmi sa aktuálny prenos odlišuje od referenčných údajov.

Payload tunel

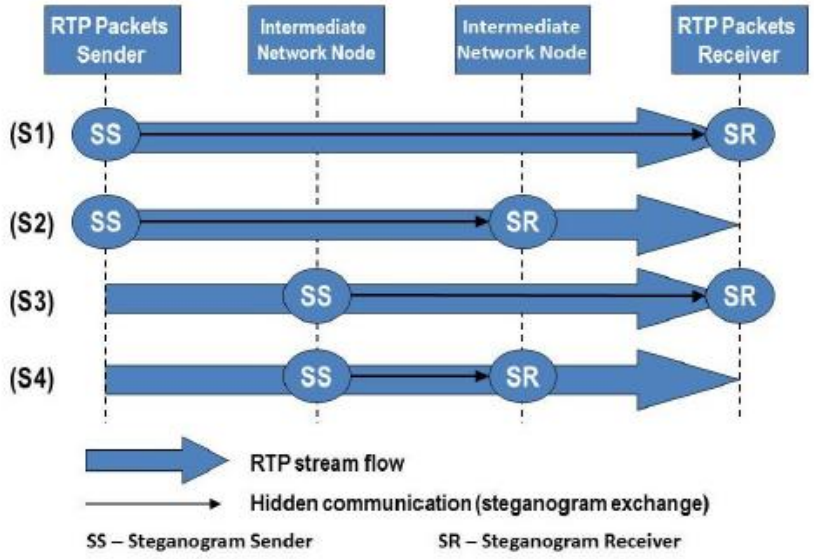
Jedná sa o vloženie dátovej komunikácie do iného protokolu. Táto možnosť je využiteľná v prípade blokovania komunikácie iných ako povolených protokolov. Napríklad použitie iba HTTP alebo emailových protokolov (SMTP, POP, IMAP). Jedná sa o echo requesty. Podobne je možné tunelovať SSH cez HTTP alebo obecne UDP a TCP cez HTTP.

Detekcia protokolov uložených v iných protokoloch je najčastejšie založená na finger printoch. Vytvorí sa databáza, do ktorej sa uložia rôzne metriky pre daný protokol a následne sa komunikácia porovná s databázou a zistia sa odchýlky.

TranSteg

Špecifický typ nahradenia payloadu navrhol Mazurczyk et al. [20]. Jedná sa o nový steganografický spôsob použitia skrytých kanálov TranSteg (Transcoding Steganography). Tento spôsob sa dá použiť v rôznych real-time aplikáciách. Požaduje aby sa prenášali skomprimované dáta, nezáleží či skomprimované stratovo alebo bezstratovo. TranSteg ukazuje použitie na IP telefónii. Hlavná myšlienka spočíva v nahradení hlasu v payloade paketov. Ten je nahradený hlasom, ktorí je lepšie skomprimovaný, aby zabral menej miesta. Zbytok, ktorý zostane po ušetrení miesta, sa môže vyplniť ľubovoľnými dátami. TranSteg sa snaží použiť kodek hlasu, ktorý zaberie menej miesta ale na kvalite by to nemalo byť poznať. Tento spôsob je možné použiť aj na šifrovaný prenos hovoru. Zabezpečenie vykonáva SRTP (Secure Real-time Transport Protocol).

Autori algoritmu ukazujú vlastnosti TranSteg na troch parametroch: *Steganographic bandwidth* (veľkosť dát prenesených za časovú jednotku), *undetectability* (schopnosť skryť samotný prenos skrytých dát), *steganographic cost* (degradácia prenosu kvôli skrytému kanálu, v tomto konkrétnom prípade sa jedná o mieru zhoršenia kvality hovoru a vznik zdržania kvôli prekódovaniu hovoru). Steganographic bandwidth súvisí s veľkosťou prenášaných dát, resp. použitým kodekom. Čím je kompresia menšia tým sa dá zakódovaný hlas zmenšiť a potom vznikne viac miesta na skryté dáta. Úspešnosť aplikácie algoritmu záleží aj na pozícii vkladateľa skrytých dát. Ten môže byť na rôznych miestach (viď obrázok 4, ktorý je analógiou k obrázku 3).



Obrázok 4 Pozície odosielateľa a príjemcu. Zdroj TranSteg [20]

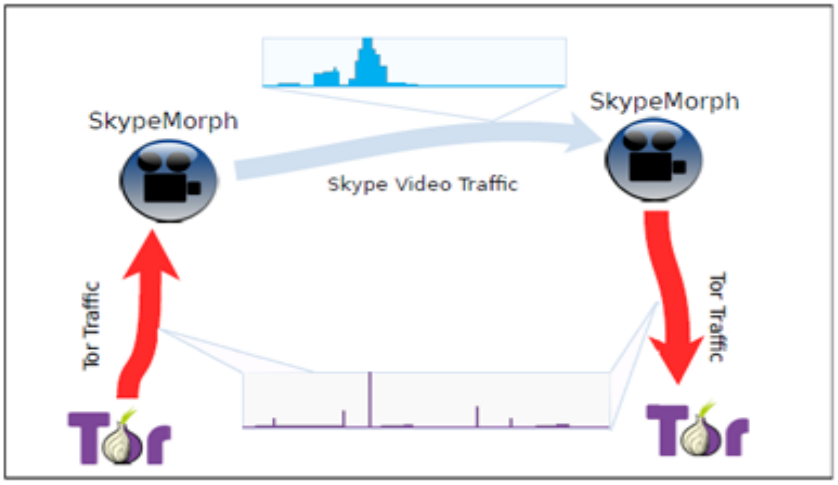
Prvý prípad S1 je najvhodnejší. Odosielateľ skomprimuje hovor silnejším kodekom a vzniknutý priestor doplní tajnými dátami. V tomto prípade sa nekoná žiadne prekódovanie hovoru. Spolu s S2 je tu možnosť u odosielateľa vybrať kodek, ktorý je vhodný na prenos. Naviac u S2 príjemca musí prekódovať hovor na pôvodný a ten poslať ďalej. Prípady S3 a S4 majú nevýhodu, že odosielateľ už nemôže vybrať kodek a preto sa musí snažiť, čo najvhodnejšie vložiť dáta už do prenosu. Preto nie je zaručený dobrý steganographic bandwitdh. Použitie šifrovania SRTP nie je problém ani v prípadoch iných ako S1, pretože vždy sa prenesú kľúče ešte v nezašifrovanej podobe pred začiatkom zabezpečenej komunikácie.

Problém detekcie skrytého kanálu závisí na pozícii kontrolóra. Ak má možnosť prístupu iba v jednom bode na siete, tak by nemal nič spozorovať. Pretože aj po pridaní skrytých dát má prenos rovnaké vlastnosti ako bežný VoIP (Voice over IP). Problematickejšia je možnosť získať dáta na dvoch miestach. V prípade S2-4 by kontrolór porovnal dáta pred a po kódovaní a zistil by odchýlky. Steganographic cost závisí od kodeku. Niekedy sa nepodarí zakódovať veľa tajných dát, pretože hlas

už nemôže byť skomprimovaný o veľa lepšie. Ale zdržanie pri prekódovaní je veľmi malé a preto by malo zostať nepovšimnuté.

SkypeMorph

Zamaskovanie dát jedného protokolu za iný a súčasné skrytia dát pomocou siete Tor poskytuje technika SkypeMorph [21]. Pripojenie do siete Tor je možné cez vstupné body, ktoré sú zverejnené na internete alebo cez špeciálny typ vstupných bodov bridge. Ale aj oni sú poskytované na internete, ale v obmedzených množstvách. Tým je možné tieto adresy zaradiť na zoznam blokovaných adries a zamedziť použitie Toru. Tomu sa snaží SkypeMorph (schéma aplikácie sa nachádza na obrázku 5). zabrániť. Komunikáciu k vstupnému bodu zamaskuje ako Skype video hovor.



Obrázok 5 Schéma aplikácie SkypeMorph. Zdroj SkypeMorph[21]

Tor za skladá z niekoľkých prvkov. *Directory servers*, k ním sa užívatelia pripájajú, aby získali zoznam uzlov, cez ktoré budú prenášať dáta. Prenosové uzly sa nazývajú *onion routers*. *Bridge* slúži na zamedzenie zabránenia prístupu k týmto serverom. Sú to neverejné prístupové body do siete, na vyžiadanie sú užívateľovi poskytnuté adresy, na ktoré sa môže pripojiť. Aj napriek nezverejneniu ich kompletného zoznamu, táto technika nie je úplne neprekonateľná.

Skype je uzavretý protokol, aj tak autori uvádzajú niektoré jeho vlastnosti. Správy sú zabezpečené šifrovacím algoritmom AES. Na autentizáciu sú používané RSA certifikáty. Tiež sa podarilo zistiť, že Skype používa nejaký druh zaistenia integrity. Preto jediné spôsoby narušenia komunikácie je úmyselné zahodenie dát (packet dropping) alebo odopretie služby (Denial of service). Skype sa skladá z troch typov uzlov. Servery sa starajú o autentizáciu pri prihlasovaní. Normálne uzly sú časti P2P komunikácie. Tretia možnosť sú superuzly. Disponujú kvalitnou konektivitou a slúžia ako premostenie komunikácie pre užívateľov za firewallom alebo NATom. Samotný prenos dát sa deje cez UDP, ale môže sa vytvoriť aj TCP spojenie pre signalizáciu. Skype je schopný pracovať s viacerými kodekmi pre hlas a video aby sa lepšie vedel prispôsobiť sieti.

Samotná aplikácia SkypeMorph sa skladá z dvoch častí . Serverová pracuje tak, že najprv sa prihlási do Skype siete a vytvorí si UDP port na neskoršiu komunikáciu. Klientska časť sa tiež pripojí do Skype siete. Cez vytvorený UDP port sa pripojí k serveru a spárujú sa. Následne klient zaháji Skype video hovor so serverom. Ten po určitom časovom intervale, rádovo v sekundách, ukončí. Následne nastáva samotná maskovaná Tor komunikácia, na ktorú sú použité UDP porty. Komunikácia sa snaží napodobiť vlastnosti Skype protokolu. Hlavne sa jedná o veľkosti paketov a časy ich odoslaní.

Zhrnutie skrytých kanálov

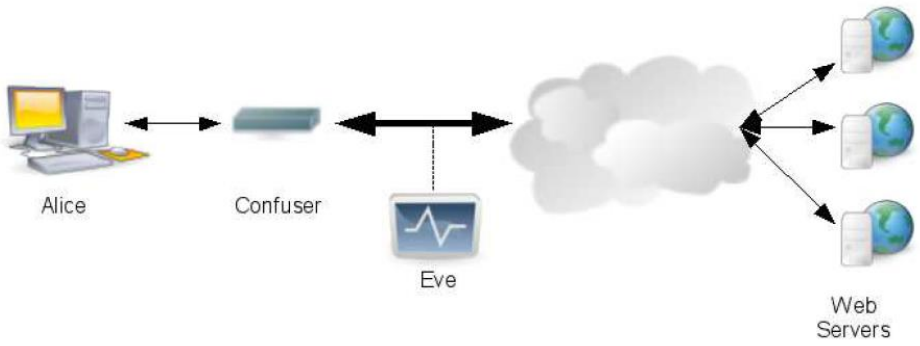
Toto boli možné použitia skrytých kanálov. Napriek popisu len niektorých druhov je vidieť, že možností je dostatok. Ale ak o nich odposluch vie, dá sa im zamedziť, prípadne obmedziť v rozumnej miere. Ich detekcia je najväčší problém. Nikdy sa nedá stopercentne všetko úplne zabezpečiť. Preto sa dá súhlasiť s názorom Zander et al. [6]. Ideálne je vyhnúť sa skrytým kanálom v *design* fáze. Pri návrhu systému sa treba čo najviac zamyslieť nad jeho používaním a nad jeho vlastnosťami. Snažiť sa zistiť čo môžu byť slabé miesta, ktoré by mohli viesť k obídeniu bezpečnostnej kontroly.

3.2 Zmätenie odposluchu

Ďalší spôsob skrytia dát je možnosť generovať nadbytočné dáta, ktoré sa pošlú spolu s normálnymi a tak znížia možnosť odhalenia. Cronin et al [22] prezentuje niektoré postupy ako to dosiahnúť.

Spôsob skrytia dát spočíva v tom, že medzi komunikáciu užívateľa a serveru (napríklad web server) sa vloží prvok, slúžiaci na zmätenie odposluchu (môže sa jednať o počítač alebo nejaký aktívny sieťový prvok). Tento prvok bude generovať falošné HTTP Requesty na server. Môžu to byť hocijaké požiadavky na prijatie dát. Odposluch sa bude nachádzať medzi prvkom a serverom (situácia na obrázku 6). Tým, že prvok je medzi zdrojom skutočných dát a užívateľom, môže Requesty na dáta vytvárať ale aj zároveň požadované dáta prijímať. Dôležité je, že prijaté dáta nemusí posielať ďalej. Práve on príjme odpovede na vymyslené odoslané požiadavky a zahodí ich. Užívateľ nemusí vedieť, že prenos vymyslených dát existuje. Ale pravdepodobne to vie, pretože sa jedná o utajenie jeho komunikácie. Možná nevedomosť tejto skutočnosti sa nazýva *plausible deniability*. Výhodou je aj to, že odposluch nevie, ktoré dáta sú skutočne od A a ktoré od mätúceho prvku. Aj v prípade zachytenia veľkého počtu dát je možné, že A neposlalo po sieti vôbec nič.

Toto je ukázané na postupe použitia vo Wi-Fi sieťach (situácia na obrázku 6). V obyčajných ethernetových sieťach, ktoré sú správne nakonfigurované (nepoužitie hubu a zamedzenie preťaženia switchu, aby sa správal ako hub), nemôže hocikto len tak poslúchať a odchytiť komunikáciu. Toto je možné vo Wi-Fi sieťach. Všetka komunikácia sa dá zachytiť pokiaľ je útočník v dosahu. Obranou proti tomuto je zašifrovanie komunikácie. Ale aj napriek tomu autori navrhujú použitie predošlého postupu. Namiesto obyčajného AP (Access point), sa použije Confusion Access Point (CAP).



Obrázok 6 Ukážka použitia CAP. Zdroj On the (un)reliability of eavesdropping [22]

Ten by mal byť nerozoznateľný oproti bežnému AP. Jeho funkcia spočíva v generovaní falošnej komunikácie na sieti. Tá bude pre odposluch nerozlíšiteľná od normálnej komunikácie. CAP pracuje na aplikačnej vrstve a komunikáciu simuluje na základe IP a MAC adresy skutočných účastníkov siete. CAP vytvára náhodné URL adresy (napríklad pomocou Google search). Následne CAP generuje požiadavky na prístup k týmto adresám a posiela ich do siete. Skutočný účastník o týchto dátach nič nevie a generuje TCP RST ako odpoveď. CAP tieto správy neprepošle ďalej k skutočnému serveru. A CAP tiež generuje vymyslené TCP RST pakety k skutočnej komunikácii od skutočných účastníkov. Tým sa stáva vymyslená a skutočná komunikácia nerozlíšiteľná.

Tento spôsob neschová skutočné dáta, iba sa snaží vložiť veľa zbytočných dát k nim. Útočník, ktorý to zachytí, potom musí sám zistiť (väčšinou človek, lebo pre software je to dostatočne komplikované), ktoré dáta sú skutočne a ktoré tie generované. Toto pridá útočníkovi veľa práce a získa minimálne zdržanie pred odhalením dát, ktoré on posiela po sieti.

4 Návrh spôsobu ukrytia dát

Táto kapitola popisuje vybraný spôsob ukrytia dát v počítačových sieťach. Obsahuje jeho popis a príklad jeho použitia. V ďalšej kapitole sa nachádza návrh aplikácie, ktorá tento spôsob ukrytia dát využíva. V texte sa bude vyskytovať pojem cenzor. Cenzor je druh odposluchu používaný v prípadoch, ktoré sa snažia priblížiť situáciu cenzúry internetu (bez hodnotenia, či je táto činnosť správna alebo nie).

V predošlej kapitole je uvedených niekoľko rôznych spôsobov ako zabrániť prezradeniu dát z komunikácie. Tiež sú uvedené postupy ako zabrániť detekcii, či nejaká komunikácia vôbec prebieha. Zo získaných poznatkov je navrhnutý spôsob skrytia dát v počítačových sieťach.

Medzi najrozšírenejšie spôsoby používania internetu patrí návšteva internetových serverov za účelom prezerania webových stránok. Môže sa jednať o získavanie dát alebo ich odosielanie na internetový server. Okrem prehliadania bežných internetových stránok je možné pristupovať aj na emaily cez web rozhranie. Na to sa používa protokol HTTP a jeho šifrovaná verzia HTTPS. Tie sú používané ako vyššia vrstva nad protokolmi TCP a IP. Na základe získaných dát, ktoré prenášané počas komunikácie s internetovým serverom sa dá o užívateľovi veľa dozvedieť. Obmedzením takejto činnosti je možné zabrániť, aby užívateľ získaval informácie. Tiež je takto možné zabrániť predávanie určitých informácii internetovému serveru.

Kontrola a prípadné následné zamedzenie výmeny informácii sa nazýva cenzúra. V niektorých krajinách je cenzúra internetu veľmi používaná [23]. Môže byť zneužitá na kontrolu prenosu dát a následnému zabráneniu prenosu dát, ktoré sú označené ako nevhodné. Cenzor však môže takéto kontrolovanie a filtrovanie dát zneužiť. Napríklad sa neumožní prístup k informáciám, ktoré sú ideovo nesúhlasné s aktuálnou politickou situáciou. Alebo sa môže jednať o snahu zamedziť šírenie určitých informácií mimo danú krajinu. Napriek rôznym snahám o cenzúru sa ľudia snažia komunikovať a prekonať takúto informačnú bariéru. Ako príklad je možné uviesť zákaz prístupu na internetové stránky, ktoré sú v zahraničí. Alebo analogicky pristupovať zo zahraničia na internetové servery nachádzajúce sa v danej krajine.

Základná snaha o filtrovanie komunikácie môže prebiehať na základe použitých adries a portov, na ktoré sa užívateľ chce pripojiť. Ak sú určité adresy zakázané, tak jediná možnosť je pripájať sa cez nejakého sprostredkovateľa. Za predpokladu, že prístup k nemu povolený. Funkcia sprostredkovateľa môže byť realizovaná proxy serverom. Proxy server taktiež skrýva identity užívateľov pred službami, na ktoré sa pripájajú.

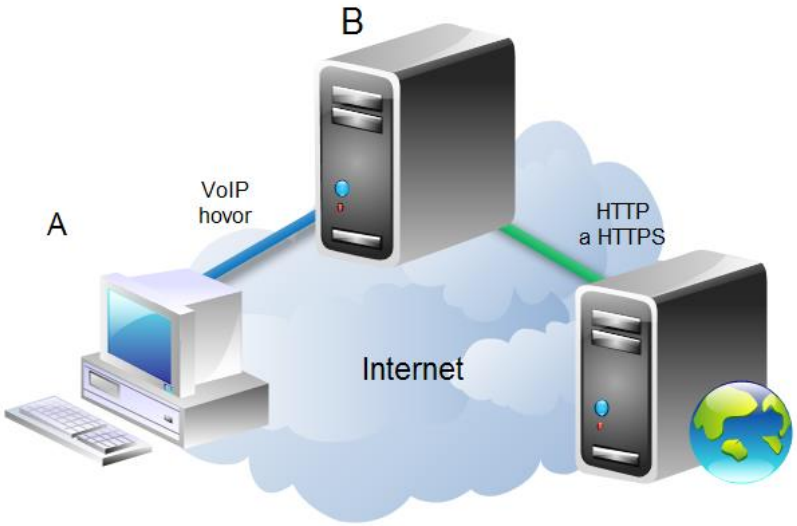
Iný spôsob je filtrovanie a prípadne neumožnenie komunikovať na základe obsahu samotných dát, ktoré putujú medzi účastníkmi takejto komunikácie [24]. Aj v prípade vytvorenia úspešného pripojenia na nejaký server, sú prenášané dáta kontrolované. Ak sú tieto dáta uznané ako nevhodné, komunikácia bude ukončená. V prípade použitia zabezpečeného protokolu HTTPS je tento spôsob

ťažší, ale tiež bežne využívaný. Strana, ktorá sa stará o filtrovanie informácii môže užívateľovi nanútiť svoj certifikát. Užívateľ má potom na výber, či ho využije alebo mu komunikácia nebude umožnená vôbec. To znamená, že komunikácia nebude ani v tomto prípade utajená pred odposluchom.

Existujú rôzne nástroje a postupy, ktoré sa snažia obísť cenzúru internetu. Ako príklad je možné uviesť Tor. Tor sieť sprostredkúva anonymné prezeranie internetových stránok. Užívateľ komunikuje priamo iba so vstupným bodom siete Tor. Funkčne sa jedná o určitú formu proxy serveru. Prenos dát v rámci siete Tor je zašifrovaný. Tiež je zaistené aby jednotlivé uzly siete nevedeli, kto je skutočným zdrojom a cieľom prebiehajúcej komunikácie. Aj keď sa jedná o prístup k internetovým stránkam, porty a protokol pre komunikáciu s Tor sieťou sú iné ako HTTP alebo HTTPS. Filtrovanie a kontrola komunikácie je v tomto prípade ťažšia. Ale má to aj svoje nevýhody. Použité porty, na ktoré sa užívateľ pripája, nemusia byť bežne používané. Komunikácia medzi užívateľom a Tor sieťou je nie celkom obvyklá a tieto fakty môžu získať pozornosť cenzora [25].

Aplikácia, ktorá je cieľom tejto diplomovej prace, sa snaží pomôcť prekonať sledovanie a prípadne následné zamedzenie komunikácie medzi užívateľom a internetovým serverom. Aplikácia sa bude snažiť skryť internetovú komunikáciu do iného dátového prenosu. Cieľom je, aby cenzor nevedel, že prebieha aj nejaká iná komunikácia. A tiež aby ani nedokázal určiť medzi kým tento prenos dát prebieha.

Tento spôsob skrytia dát využíva rozdelenie aplikácie na dve časti. Prvá časť, klientska, sa bude nachádzať u užívateľa. Druhá, serverová, niekde mimo dosah cenzora. Cenzor má prístup ku úplnej komunikácii medzi klientskou a serverovou časťou. Ale už nemá prístup k dátam prenášaným medzi serverovou časťou a internetom.



Obrázok 7 Schéma skrytia dát

Na obrázku 7 počítač A predstavuje užívateľa, ktorý komunikuje s internetovým serverom. Počítač B predstavuje serverovú časť aplikácie. Na počítači A sa spustí klientska časť aplikácie a bude sprostredkovávať komunikáciu užívateľa s internetom. Medzi A a B bude komunikácia vyzerať ako VoIP hovor. Môže sa jednať o audio hovor. Ale výhodnejšie bude, aby sa komunikácia tvárila ako video hovor. Pretože pri video hovore je prenášané väčšie množstvo dát. Preto môže byť aj skrytá komunikácia dátovo náročnejšia. A následne medzi B a internetom bude prebiehať pôvodná komunikácia typu HTTP alebo HTTPS.

Obrázok 8 Formát dátového prenosu

Medzi klientskou a serverovou časťou bude po celý čas prebiehať falošný VoIP hovor. To bude pozorovať cenzor (komunikácia medzi bodmi A a B na obrázku 8). V skutočnosti to bude zamaskovaná komunikácia medzi užívateľom a internetovým serverom. Konkrétne sa bude jednať o tunelovanie protokolov HTTP a HTTPS do protokolu RTP [26]. Podmienka, aby to fungovalo je, že musí byť umožnené vytvoriť VoIP hovor medzi klientskou a serverovou časťou.

Z pohľadu užívateľa skrytie dát nebude vôbec badateľné a nebude ničím obmedzovaný. Užívateľ nastaví v internetovom prehliadači použitie proxy serveru – ten reprezentuje klientska časť aplikácie. Užívateľ bude len vedieť, že komunikácie medzi prehliadačom a internetovým serverom prechádza cez proxy server.

Úlohou klientskej časti je pretransformovať prijatú komunikáciu, ktorú získa z prehliadača do interného protokolu aplikácie. Následne tieto dáta preposlať do serverovej časti. Tam budú dáta extrahované. A následne predané požadovanému cieľu. Cieľom komunikácie je internetový server, s ktorým bude komunikácia prebiehať HTTP a HTTPS protokolom. Internetový server nebude vedieť, že dáta boli po ceste transformované a že skutočný zdroj nie je serverová časť aplikácie. Z jeho pohľadu to bude úplne normála komunikácia. Tento spôsob skrytia dát je pre užívateľa a aj internetový server úplne transparentný. Ani zdroj a ani cieľ vôbec nevedia, že komunikácie medzi nimi je nejakým spôsobom menená a tunelovaná prostredníctvom iného protokolu – VoIP hovoru.

Účel tohto spôsobu skrytia dát je nasledovný:

Z pohľadu zdroja (užívateľa) a cieľa komunikácie (internetového serveru)

- je to skrytie skutočnej identity užívateľa. Je to možné prirovnať ku bežnému proxy serveru alebo exit nodu v sieti Tor.

Z pohľadu odposluchu (má k dispozícii všetky dáta prenesené medzi klientskou a serverovou časťou)

- odposluch nevie s kým užívateľ komunikuje

- odposluch nevie aké dáta sa prenášajú
- odposluch nevie, že nejaká komunikácia pomocou protokolu HTTP alebo HTTPS (taktiež i TCP) vôbec prebieha

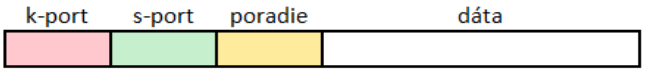
Za predpokladu, že odposluch sleduje len výskyt HTTP protokolu a HTTPS protokolu, prípadne TCP protokolu, tak sa z jeho pohľadu žiadna komunikácia neprebieha. Odposluch môže iba vidieť, že prebieha VoIP hovor medzi užívateľom a serverovou časťou aplikácie.

Tento spôsob skrytia dát sa však musí snažiť čo najviac priblížiť bežnému VoIP hovoru, aby bolo šanca odhalenia čo najmenšia.

Hlavný rozdiel medzi prehliadaním internetových stránok a VoIP telefóniou je, že VoIP hovor prebieha na UDP a nie na TCP. A taktiež veľkosť paketov a časy ich príchodov a odchodov sú iné než v prípade VoIP hovoru. VoIP hovor odosiela dáta v pravidelných časových intervaloch (v prípade niektorých kodekov sú všetky pakety rovnako veľké). Pri prehliadaní internetových stránok sú dáta zasielané nepravidelne (napr. pri načítaní novej internetovej stránky) a majú rôznu veľkosť. Ďalší fakt je, že navrhovaná aplikácia sa musí vysporiadať s dobou, počas ktorej nie sú prenášané žiadne dáta (užívateľ si prezerá internetovú stránku a v tom okamihu žiadne dáta nie sú prenášané).

Samotný spôsob skrytia dát v počítačových sieťach teda funguje nasledovne: Medzi klientskou a serverovou časťou sa na začiatku vymenia SIP [27] správy. Tie fungujú ako skutočný pokus o nadviazanie hovoru VoIP. Slúžia na to aby bol spôsob vierohodný a ozrejmil, že ďalej budú stále chodiť RTP pakety medzi klientom a serverom. Pretože skutočný VoIP hovor prebieha neustále, tak medzi klientsku a serverovou časťou sú stále posielané pakety. Oni simulujú prenos audio alebo video. Jedná sa o pakety RTP protokolu. Majú správne hlavičky (z hľadiska špecifikácie protokolu) a aj veľkosť ako keby sa jednalo o skutočný VoIP hovor. Časy ich odoslania sa tiež snažia simulovať skutočný prenos dát VoIP hovoru. Rozdiel oproti VoIP hovoru je multimediálna časť týchto paketov. Payload je v tomto prípade tvorený dátami protokolov HTTP a HTTPS. Alebo náhodne zvolenými dátami, ak žiadne dáta protokolov HTTP a HTTPS nie sú v danom časovom okamžiku k dispozícii.

Klientska a serverová časť majú za úlohu dáta tunelovaných protokolov prispôsobiť, aby ich bolo možné vložiť ako paypload do RTP. Ich úlohou je predovšetkým rozkúskovať ich tak, aby sa do RTP payloadu zmestili, a tiež kontrolovať či sú úspešne prijaté druhou stranou. Pred tieto dáta sú ešte vložené interné hlavičky aplikácie. Obsahujú metadáta, ktoré sa starajú o úspešnosť tohto prenosu medzi klientom a serverom. Forma prenosu dát interného protokolu je zobrazená na obrázku 9.



Obrázok 9 Enkapsulácia dát v internej komunikácii

Popis enkapsulácie dát z obrázku 9:

- k-port označuje číslo portu klientskeho procesu
- s-port označuje číslo portu serverového procesu
- poradie je sekvenčné číslo paketu
- dáta sú prenášané dáta (HTTP a HTTPS komunikácia)

Nasleduje detailnejší popis priebehu komunikácie medzi užívateľom a skutočným cieľom komunikácie.

1. Po spustení aplikácie bude nasledovať prenos SIP správ, ktoré inicializujú VoIP hovor.
2. Následne už neustále prebieha prenos RTP paketov medzi klientskou a serverovou časťou. Ak nie sú k dispozícii žiadne zdrojové dáta, tak RTP pakety obsahujú náhodne zvolené dáta. Takéto pakety sa po prijatí hneď zahadzujú.
3. Užívateľ posiela prostredníctvom internetového prehliadača požiadavky na dáta ku klientskej časti. Ta získa cieľovú adresu a port (podľa toho či sa jedná o protokol HTTP alebo HTTPS). Taktiež sa získa port, z ktorého požiadavka prišla. To je dôležité, aby sa odpoveď zaslala spať správnemu procesu. Klient takto získané dáta rozdelí na menšie časti, ak je to potrebné kvôli ich veľkosti. Nasledovne ich opatrí internou hlavičkou. Následne sa priloží RTP hlavička. Takýto paket je zaslaný na serverovú časť. Ta odstráni RTP hlavičku. Následne predá požiadavku skutočnému cieľu komunikácie, konkrétne internetovému serveru. Pomocou interného protokolu vznikne väzba medzi klientskou časťou a serverovou časťou. Toto je dôležité, aby sa komunikácia rôznych požiadaviek nepomiešala. Pretože naraz môže prebiehať viac samostatných spojení protokolov HTTP alebo HTTPS. Po tom ako serverová časť získa odpoveď od internetového serveru, postup sa spätne opakuje. Serverová časť pošle dáta do klientskej, tá ich extrahuje a predá internetovému prehliadaču. Spojenie sa po prenose dát neukončí. Ukončí sa iba v prípade, že koniec spojenia inicializuje prehliadač alebo internetový server. Spojenie sa neukončuje aby mohlo byť použití viackrát.

V tejto komunikácii sú špeciálne dva príklady: nadviazanie spojenia s internetovým serverom a ukončenie spojenia s internetovým serverom.

Nadviazanie spojenia: Klientsky proces čaká na prijatie požiadavky od internetového prehliadača. Po jeho prijatí klient vytvorí proces, ktorý bude spracovávať toto spojenie. Z prvej požiadavky získa adresu cieľu a tiež informáciu o použitom protokole. Podľa toho sa zistí či sa jedná o HTTP alebo jeho šifrovanú verziu HTTPS. Túto úvodnú informáciu zašle serverovej časti. Tá tiež vytvorí nový proces a vytvorí spojenie k internetovému serveru. Ak sa podarí pripojiť, tak nasledovná komunikácia prebieha normálne. V prípade, že sa pripojiť nepodarí (môže sa jednať o neexistujúcu adresu atď.) tak serverová časť odošle informáciu o tomto stave klientskej časti. Klientska časť tento fakt oznámi prehliadaču.

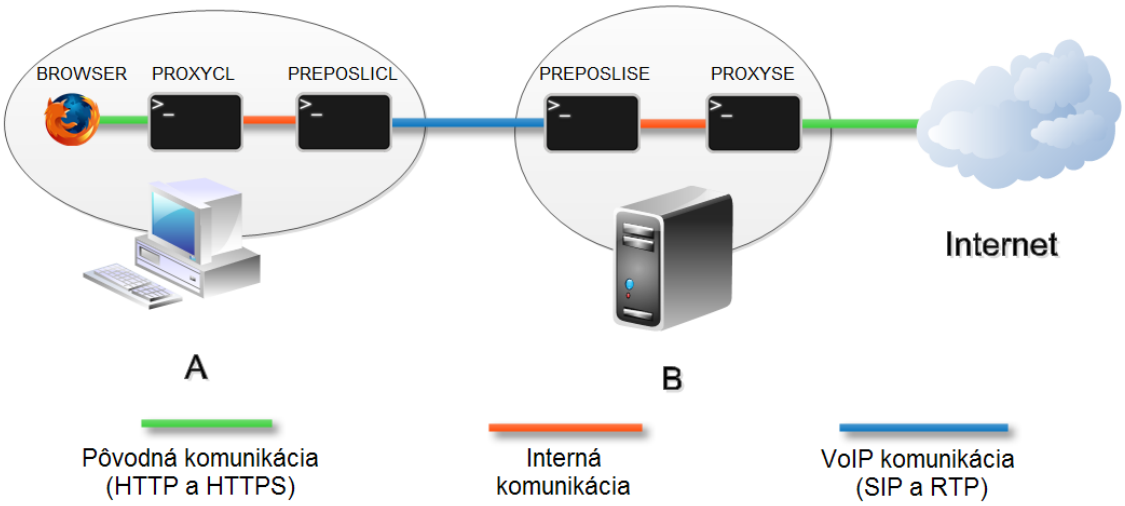
Druhý prípad je ukončenie spojenia: Ak skončí komunikácia medzi aplikáciou a užívateľom alebo aplikáciou a internetom, tak druhá strana sa to musí dozvedieť. Ak prehliadač ukončí spojenie s klientskou časťou, tak to oznámi serverovej časti. Serverová časť ukončí spojenie s internetom a následne obidva procesy skončia. V prípade ukončenia spojenia zo strany internetu sa deje to isté ale opačne. Procesy sa ukončia a uvoľnia pridelené prostriedky.

5 Návrh aplikácie

V tejto kapitole bude popísaný návrh aplikácie proxyvoip. Aplikácia bude implementovať spôsob skrytia dát v počítačových sieťach z predošlej kapitoly. Táto kapitola tiež obsahuje rozdelenie aplikácie na jednotlivé časti a ich funkcie.

Proxyvoip má úlohu prostredníka v komunikácii užívateľa s internetom. Jej úloha je maskovať komunikáciu HTTP a HTTPS protokolmi za VoIP hovor. Nasledovný obrázok 10 znázorňuje jednotlivé časti aplikácie. Jedná sa o spresnenie návrhu komunikácie z predošlej kapitoly a obrázku 7.

Aplikácia proxyvoip sa skladá zo samostatných programov: *proxycl*, *proxyse*, *preposlicl* a *preposlise*. Tieto programy vytvárajú dve časti aplikácie. Klientsku časť tvoria dva programy: *proxycl* a *preposlicl*. Serverová časť pozostáva tiež z dvoch programov: *preposlise* a *proxyse*. Na obrázku 10 sú znázornené programy klientskej časti na jednom počítači a programy serverovej časti na druhom.



Obrázok 10 Schéma návrhu aplikácie proxyvoip

Z obrázku 10 je vidieť časti, z ktorých sa aplikácia proxyvoip skladá:

- Klientska časť – na počítači A
- Serverová časť – na počítači B

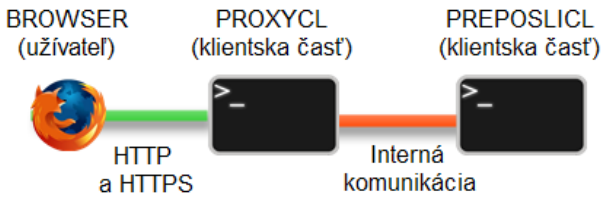
Programy *proxycl* a *proxyse* majú na starosti komunikáciu s užívateľom a aplikáciou resp. aplikáciou a internetom. Prijímajú od nich dáta alebo im dáta dodávajú. Na komunikáciu medzi *proxycl* a *proxyse* slúžia programy *preposlicl* a *preposlise*. *Preposlicl* a *preposlise* slúžia na vytvorenie a spravovanie falošného VoIP hovoru.

5.1 Popis jednotlivých častí aplikácie

Klientska časť aplikácie slúži na komunikáciu medzi internetovým prehliadačom a serverovou časťou. Jej úloha je transformovať požiadavky a dáta prijaté z prehliadača do internej formy. Takto upravené dáta následne zamaskovať ako RTP pakety a tie následne odoslať serverovej časti. Klientska časť takisto slúži na prijatie dát zo serverovej časti prostredníctvom programov *preposlicl a preposlise*. Po ich prijatí ich prevedie z internej formy do pôvodnej. Tieto dáta následne predá prehliadaču.

V nasledovnom texte bude bližšie opísané ako ktoré časti aplikácie proxyvoip fungujú. V prípade, že sa bude popis vzťahovať na obidva programy *proxycl a proxyse*, bude v texte uvedený len pojem *proxy*. Taktiež v prípade odkazovania na programy *preposlicl* a zároveň *preposlise*, bude odkazované na programy *preposli*.

Podrobnejší popis programov klientskej časti:



Obrázok 11 Pozícia proxycl

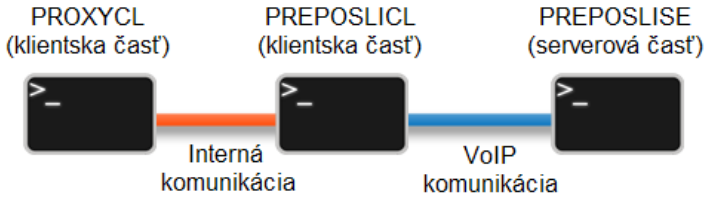
- **Proxycl**: tento program slúži na komunikáciu medzi samotnou aplikáciou a prehliadačom (viď obrázok 11). Po spustení si vytvorí soket a na tom poslúcha prichádzajúce spojenia od prehliadača. Prehliadač tento program vníma ako bežný proxy server a preto všetku komunikáciu posiela k nemu. Z hľadiska štruktúry celej aplikácie má *proxycl* na starosti komunikáciu medzi prehliadačom a programom *preposlicl*.

  Proxycl po počiatočnom nastavení soketu beží v nekonečnej smyčke a čaká na prichádzajúce spojenia od prehliadača. Po prijatí takéhoto spojenia sa vytvorí nový proces, ktorý bude naďalej pracovať s aktuálnym spojením s prehliadačom. Hlavný proces bude znovu čakať na nové pripojenie.

  Novo vytvorený proces príjme prvé dáta z prehliadača. Na základe nich zistí či sa bude jednať o HTTP alebo HTTPS. Následne získa aj adresu serveru, na ktorý má spojenie smerovať. Proces *proxycl* následne pošle špeciálnu správu, ktorá obsahuje process identifitier (PID) procesu, typ protokolu a adresu programu *preposlicl*. Potom čaká na odpoveď. Serverová časť aplikácie sa skúsi pripojiť na danú adresu. Ak je pripojenie úspešné, tak pošle potvrdzovaciu správu, ktorá obsahuje aj PID procesu na serverovej strane. Toto číslo sa bude vždy vkladať do internej hlavičky, aby sa na serverovej časti dáta dostali k správnemu

procesu. Následne proces *proxycl* odošle aj zbytok pôvodnej správy, ktorú prijal od prehliadača. Týmto sa skončila úvodná fáza komunikácie.

Teraz už bude v procese bežať smyčka, ktorá príjme dáta od prehliadača a pošle ich cez *preposli* k serverovej časti alebo príjme dáta z *preposlicl* a tie predá prehliadaču. Toto sa bude vykonávať až pokým niektorá strana neukončí spojenie. Ak prehliadač spojenie ukončí, proces *proxycl* pošle špeciálnu internú správu k serverovej časti. Serverová časť tiež ukončí spojenie s internetovým serverom. Následne sa obidva procesy aj na klientskej a serverovej časti ukončia. V prípade, že spojenie je ukončené z internetovej serveru, tak sa analogicky ukončí spojenie s prehliadačom a následne sa ukončia procesy *proxy*.



Obrázok 12 Pozícia preposlicl

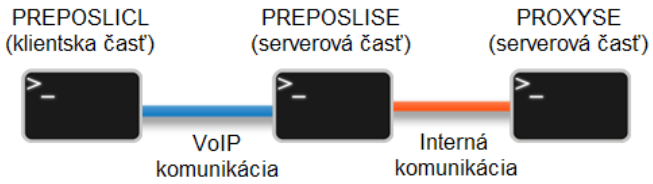
- **Preposlicl**: táto časť aplikácie slúži na simulovanie VoIP hovoru medzi klientskou a serverovou časťou aplikácie. V rámci aplikácie zastáva miesto medzi *proxycl* a *preposlise* (viď obrázok 12). Program *preposlicl* sa po spustení snaží vytvoriť o nadviazanie VoIP hovoru s protistranou, ktorá je tvorená programom *preposlise*. Na toto sa využijú správy SIP. Ak takéto nadviazanie spojenia prebehne v poriadku, nastáva hlavná fáza programu. Počas nej sú neustále posielané RTP pakety do *preposlise*. Pomocou náhodne generovaného čísla sa na 60 percent bude jednať o poslanie audio paketu a 40 percent video. Časy odoslania každého paketu budú tiež zvolené náhodne. Bude sa jednať o časy medzi 15 až 25 ms. Telo RTP paketov bude vyplnené náhodnými dátami. O náhodné veličiny sa postará generátor pseudo náhodných čísel.

  V prípade, že *preposlicl* príjme nejaké dáta od preposlise, pokúsi sa zistiť či sú to skutočné dáta alebo iba náhodné dáta určené na simulovanie komunikácie. Ak sa jedná o skutočné dáta, tak ich následne prepošle do *proxycl*. Ak to budú náhodné dáta simulujúce VoIP hovor, tak po ich prijatí už s nimi nič robiť nebude. Na rozlíšenie toho sa použije pridaná hlavička udávajúca veľkosť skutočných dát v RTP pakete. Tá sa pridáva ešte naviac k internej hlavičke aplikácie proxyvoip. Slúži iba na komunikáciu medzi *preposlicl* a *preposlise*. V RTP pakete môžu byť okrem skutočných dát prítomné ešte ďalšie náhodne generované dáta. To sa deje z dôvodu priblíženia sa veľkosti paketov skutočnému RTP paketu. Do *proxycl* sú odoslané iba skutočná dáta. Ak hlavička obsahuje veľkosť skutočných

paketov udávanú číslom nula, v tom prípade je celý paket tvorený iba náhodnými dátami a nič sa do *proxycl* neposiela.

V prípade prijatia dát od *proxycl* sa prijaté dáta spracujú a odošlú do *preposlise*. Spracovanie dát je vykonávané analogicky ako pri prijatí dát od *preposlise*. Spočíva v pridaní hlavičky udávajúcej veľkosť skutočných dát. Toto je potrebné, pretože veľkosť dát od *proxycl* môže byť menšia ako bežné dáta v RTP paketoch. Preto sa tie skutočné dáta podľa potreby doplnia náhodnými a až následne je paket odoslaný.

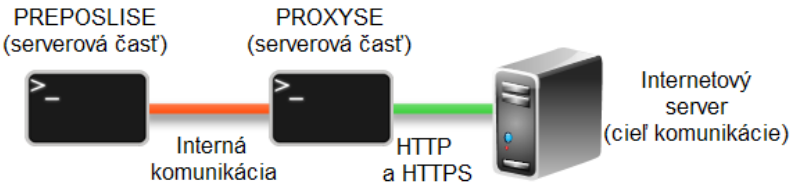
V prípade ukončenia aplikácie sa prenos ukončí tak ako sa ukončuje skutočný VoIP hovor. *Preposlicl* a *preposlise* si medzi sebou vymenia SIP správy, ktoré znamenajú korektné ukončenie hovoru.



Obrázok 13 Pozícia preposlise

- **Preposlise**: program má na starosti komunikáciu medzi *preposlicl* a *proxyse* (viď obrázok 13). Po spustení čaká na počiatočnú správu od *preposlicl*. Na základe jej prijatia nastane výmena SIP správ, ktoré znamenajú začiatok VoIP hovoru.

  Následne si *preposlicl* a *preposlise* medzi sebou posielajú RTP pakety simulujúce VoIP hovor. Pakety sú posielané buď s náhodnými dátami alebo so skutočnými. *Preposlise* funguje principiálne rovnako ako *preposlicl*. To znamená, že dáta získané od *proxyse* transformuje na RTP pakety a tie pošle *preposlicl*. A v prípade prijatia skutočných dát z *preposlicl*, sú dáta poslané programu *proxyse*.



Obrázok 14 Pozícia proxyse

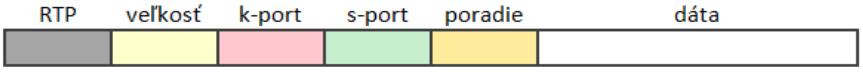
- **Proxyse**: táto časť aplikácie spravuje spojenie medzi *preposlise* a internetovým serverom (zobrazené na obrázku 14). Hlavný program beží v nekonečnej smyčke. Čaká na prijatie počiatočnej správy od *proxycl*, ktorá je doručená prostredníctvom programu *preposlise*.

Následne vytvorí nový proces, ktorý sa bude starať o konkrétne spojenie. Novo vytvorený proces získa z počiatočnej správy adresu, na ktorú sa chce užívateľ pripojiť. *Proxyse* skúsi vytvoriť spojenie na danú adresu a daný port. Ak sa mu to úspešne podarí, informuje o tom zaslaním správy do *proxycl*, opäť prostredníctvom programov *preposli*. V prípade neúspešného spojenia s internetovým serverom, program *proxyse* o tom informuje *proxycl* a následne sa ukončí. Po inicializácii pripojenia k internetovému serveru, ktoré je úspešné, program beží v nekonečnej smyčke. Prijíma dáta od internetu a tiež od *preposlise*. V prípade prijatia dát z internetu *proxyse* pridá dátam internú hlavičku a odošle ich do *preposlise*. V opačnom prípade, ak *proxyse* príjme dáta z *preposlise*, odstráni ich internú hlavičku a odošle ich internetovému serveru. Táto činnosť je vykonávaná neustále. Až pokiaľ nenastane koniec spojenia. Spojenie môže byť ukončené zo strany internetového serveru. V tomto prípade túto udalosť *proxyse* oznámi *preposlise* a následne sa ukončí. Ak *proxyse* príjme správu od *preposlise* o ukončení spojenia zo strany prehliadača, ukončí spojenie s internetovým serverom a následne sa tiež ukončí.

5.2 Popis formátu dát aplikácie proxyvoip

Všetky dáta, ktoré vstúpia do aplikácie sa postupne upravia do internej podoby. HTTP a HTTPS dáta, vstupujúce do proxyvoip z prehliadača alebo z internetového serveru majú rôznu veľkosť. Preto je nevyhnutné upraviť ich veľkosť, aby sa mohli poslať zamaskované ako RTP pakety VoIP hovoru.

Dáta do aplikácie proxyvoip vstupujú prostredníctvom programov *proxy*. Tieto programy dáta načítajú a pripravia na odoslanie do programov *preposli*. Prvý krok je obmedzenie veľkosti jednotlivých paketov. Príchodzie dáta sa rozdelia a následne im je pridaná interná hlavička (Viď obrázok 15. V rámci programov *proxy* sa ešte nepoužije pole veľkosť. Toto pole hlavičky sa pridá až pri komunikácii programov *preposli*. Pole RTP na obrázku znamená hlavička protokolu RTP a nie je súčasťou internej hlavičky.). Pole poradie slúži na spätnú rekonštrukciu pôvodných dát. Takto rozkúskované dáta s internou hlavičkou sú predané programom *preposli*. Na obrázku 15 sa tiež vyskytujú polia hlavičky *k-port* a *s-port*, tie budú vysvetlené v nasledujúcej podkapitole.



Obrázok 15 Štruktúra paketu zaslaného medzi klientskou a serverovou časťou

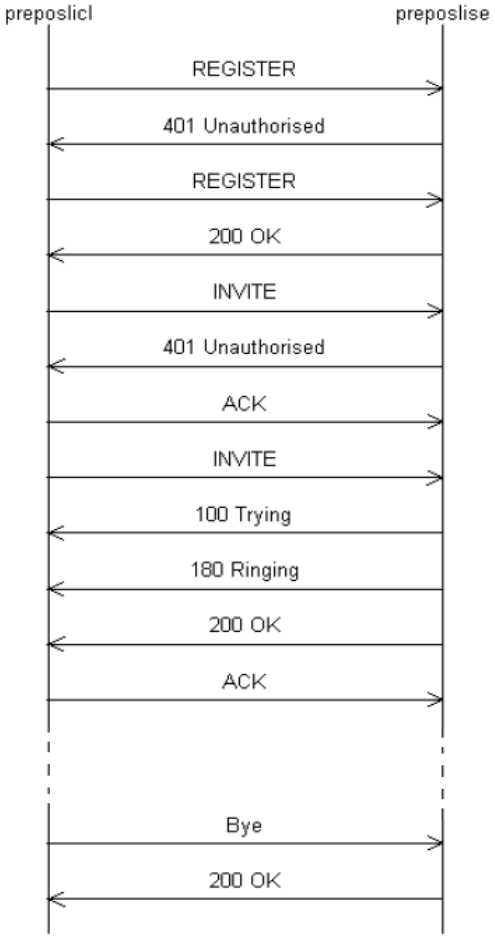
Preposlicl alebo *preposlise* dáta prijmú. Ku internej hlavičke pridajú ešte jedno pole. Pole *veľkosť* udáva veľkosť pôvodných dát spolu s veľkosťou internej hlavičky v pakete. Následne sa tieto dáta doplnia ešte ďalšími náhodne zvolenými dátami. Tak aby veľkosť paketu bola podobná veľkosti

skutočných RTP paketov. Následne sa už iba pridá RTP hlavička (prvá časť paketu na obrázku 15) a paket sa odošle z *preposlicl* do *preposlise* alebo opačne.

Následne nastáva presne opačný proces. *Preposlicl* resp. *preposlise* prijmú paket. Načítajú hlavičku udávajúcu veľkosť skutočných dát vrátane internej hlavičky. Do *proxycl* alebo *proxyse* pošlú iba skutočné dáta. V *proxycl* resp. *proxyse* sa z prijatých dát odstráni aj interná hlavička. Následne sa počká pokiaľ príde celý pôvodný paket a ten sa potom odošle internetovému serveru resp. prehliadaču.

5.3 Popis komunikačného protokolu aplikácie proxyvoip

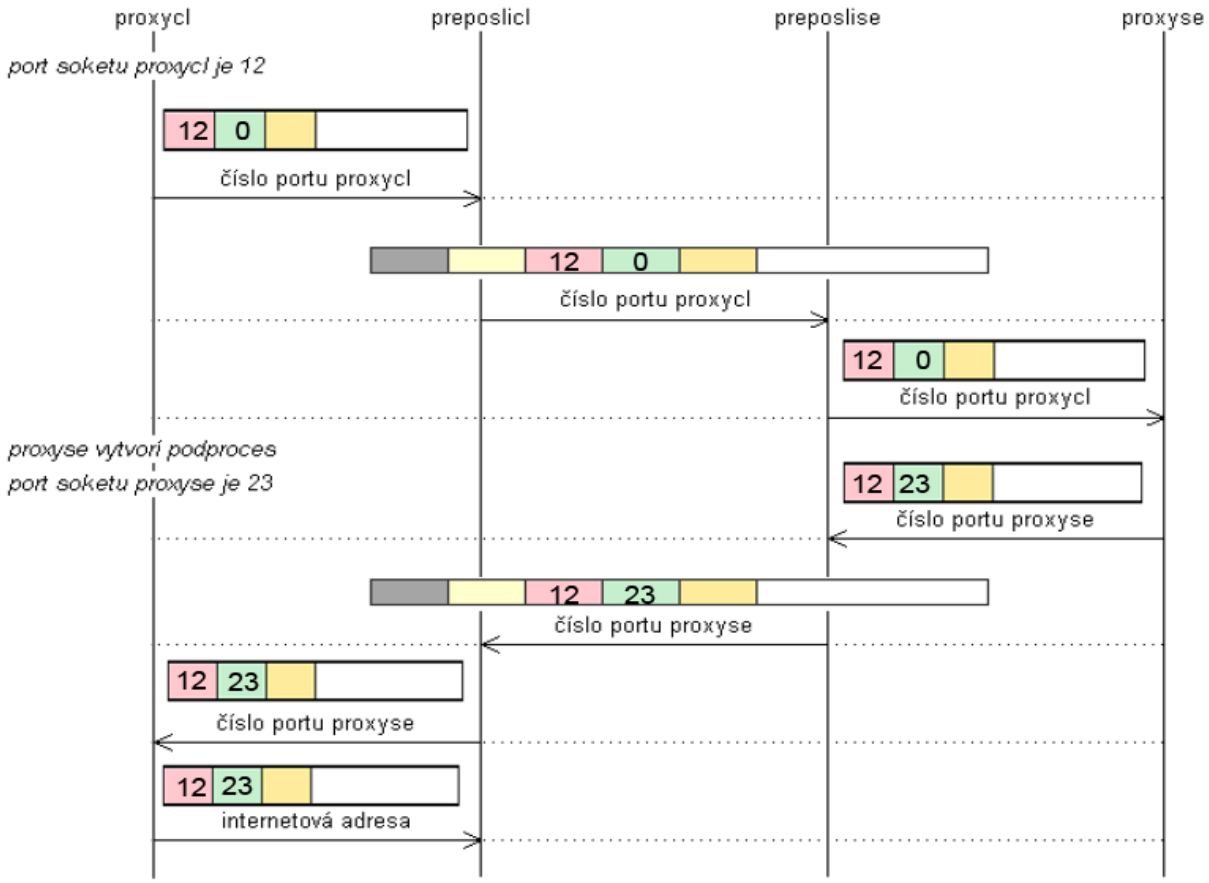
Aplikácia proxyvoip používa ku svojej činnosti signalizačné správy, internú hlavičku a falošnú SIP komunikáciu.



Obrázok 16 SIP správy znázorňujúce vytvorenie a ukončenie hovoru

- **SIP komunikácia:** V rámci aplikácie proxyvoip nastáva komunikácia medzi *preposlicl* a *preposlise* ako prvá. Tvorí ju zaslanie správ protokolu SIP (znázornená na obrázku 16). Táto komunikácia má napodobiť situáciu vytvárania VoIP hovoru. Následne sa medzi programami *preposlicl* a *preposlise* vymieňajú RTP pakety. Štruktúra paketu je uvedená na obrázku 15. Ak sa užívateľ rozhodne aplikáciu ukončiť, medzi *preposlicl* a *preposlise* skončí komunikácia prostredníctvom RTP paketov. Tieto programy si vymenia SIP správy, ktoré znamenajú bežné ukončenie hovoru. Počiatočná SIP komunikácia a aj správy zaslané v ukončovacej fáze nemajú vplyv na samotný prenos skutočných dát. Slúžia iba na dôveryhodnejšie vytvorenie ilúzie, že užívateľ využíva na komunikáciu VoIP hovor.

- **Komunikácia medzi prehliadačom a internetovým serverom:** Pri bežnej návšteve internetových stránok prehliadač používa naraz viacero spojení, aby zvýšil rýchlosť ich načítania.



Obrázok 17 Tvorba asociácie proxycl-proxyse
(čísla portov sú v tomto prípade 12 a 23)

Z hľadiska proxy serveru to znamená, že simultánne musí poskytovať dáta pre viacero soketov. Aby dáta boli správne dodané prehliadaču, aplikácia používa asociáciu medzi *proxycl* procesom a *proxyse* procesom.

Komunikácia (počiatočná fáza komunikácie je vidieť na obrázku 17) začína programom *proxycl*, ktorý príjme požiadavku od prehliadača. Následne vytvorí proces, ktorý sa bude starať o toto jedno spojenie. Proces *proxycl* na základe prijatých dát zistí, o aký protokol sa jedná a aká je adresa serveru, na ktorom sa majú nachádzať požadované dáta. Proces *proxycl* si vytvorí soket, pomocou ktorého bude komunikovať s *preposlicl*.

Následne vloží číslo portu soketu do poľa *k-port* internej hlavičky. Číslo portu slúži ako identifikátor, ktorý určuje kam majú byť doručované požadované dáta. Dáta paketu obsahujú iba číslo portu. Pole *s-port* je doplnené hodnotou nula, pretože sa ešte nevie, ktorý proces programu *proxyse* sa bude starať o toto spojenie. Pole poradie bude nastavené na jedna. Pri každom ďalšom pakete sa zvýši o hodnotu jedna. Následne sa paket odošle do *preposlicl*. V *preposlicl* sa paketu pridá RTP hlavička a internej hlavičke sa pridá pole veľkosť. Potom sa paket odošle sa do *preposlise*. Program *preposlise* odstráni RTP hlavičku a tiež odstráni pole veľkosť internej hlavičky. Zistí hodnotu poľa *s-port* internej hlavičky. V prípade, že hodnota poľa *s-port* je nula, paket je odoslaný do programu *proxyse*. V opačnom prípade je odoslaný na soket, ktorého číslo portu je hodnota *s-port*. *Proxyse* príjme paket, vytvorí nový proces, ktorý sa bude starať o aktuálne spojenie. Novo vytvorený proces *proxyse* si vytvorí soket určený na komunikáciu s *proxycl*.

Číslo portu tohto soketu vloží do poľa *s-port* internej hlavičky a tiež do tela paketu a odošle paket do *preposlise*. *Preposlise* paket príjme, pridá RTP hlavičku a doplní internú hlavičku o pole veľkosť. Potom paket prepošle do *preposlicl*. V *preposlicl* sa odstráni RTP hlavička a pole veľkosť internej hlavičky. Na základe hodnoty z poľa *k-port* sa zistí číslo portu, na ktorý sa paket odošle. Príslušný proces *proxycl* príjme paket.

Proxycl zistí číslo portu priradeného *proxyse* procesu. Týmto sa ukončila tvorba asociácie *proxycl-proxyse*. Proces *proxycl* následne vytvorí správu, ktorá obsahuje adresu serveru na internete a príznak, o ktorý protokol sa jedná. Polia *k-port* a *s-port* internej hlavičky budú vždy vyplnené a programy *preposli* na ich základe budú smerovať pakety k príslušným procesom. Proces *proxyse* príjme správu s adresou serveru a typom protokolu. Skúsi vytvoriť spojenie k tomuto serveru. Ak sa mu to podarí bude očakávať dáta od *proxycl* alebo od internetového serveru a preposielať ich na určenému cieľu. V prípade, že sa spojenie so severom nepodarí vytvoriť, odošle sa správa o chybe do *proxycl* procesu. Ten ukončí spojenie s internetovým prehliadačom a obidva *proxy* procesy sa ukončia.

V nasledovnej fáze procesy *proxy* sprostredkúvajú komunikáciu medzi prehliadačom a internetovým serverom vzájomným preposielaním požadovaných dát. To sa deje až pokiaľ niektorá strana neukončí spojenie. Ak je spojenie ukončené na strane prehliadača, *proxycl*

proces to oznámi *proxyse* procesu, ten ukončí spojenie so serverom. Následne sa obidva *proxy* procesy ukončia. V prípade ukončenia spojenia zo strany *proxycl* sa to deje analogicky. *Proxycl* pošle správu o ukončení spojenia do *proxyse*. *Proxyse* ukončí spojenie s internetovým serverom a obidva *proxy* procesy sa následne ukončia.

6 Implementácia útoku

Táto kapitola popisuje implementáciu aplikácie proxyvoip. Bude bližšie uvedené ako jednotlivé programy fungujú a vysvetlená činnosť niektorých dôležitých funkcií.

Celá aplikácia je implementovaná v jazyku C++. Má formu konzolovej aplikácie. Aplikácia sa kladá sa zo 4 programov *proxycl, proxyse, preposlicl a preposlise*. Programy *proxycl* a *preposlicl* tvoria klientsku časť aplikácie a programy *proxyse* a *preposlise* serverovú. Všetky programy navzájom komunikujú prostredníctvom soketov. Nastavenia aplikácie proxyvoip a popis parametrov spúšťania jednotlivých programov sa nachádza v prílohe A.

Časti aplikácie proxyvoip

Program *proxycl* slúži ako vstupný bod do celej aplikácie pre internetový prehliadač. Hneď po spustení vytvorí soket, na ktorom bude čakať príchodzie spojenia. Po prijatí spojenia od prehliadača je vytvorený nový proces pomocou funkcie *fork()*. Následne sa vytvorí ďalší soket, slúžiaci na komunikáciu s programom *preposlicl*. Proces potom beží v nekonečnej smyčke *while()*. V nej prijíma dáta od prehliadača a tie odosiela do *preposlicl*. Analogicky sa to deje v opačnom smere. Materský proces *proxycl* priebežne kontroluje ukončenie svojich procesov a prijíma ich statusy. Na samotné odosielanie a prijímanie sa využívajú funkcie *odosli()* a *prijmi()*. Oni sa starajú o všetky činnosti spojené s prenosom dát.

Funkcia *odosli()* slúži na odoslanie dát. Zdrojové dáta posiela rozdelené po častiach. Zdrojové dáta sú v tejto funkcii, tak ako aj v celej aplikácii, reprezentované po znakoch kontajnerom *vector<char>*. Vector zabezpečuje dynamickú veľkosť dát a jednoduchý prístup k dátam v ňom uloženým. Funkcia *odosli()* používa pomocné funkcie, aby boli do správy vložené interné hlavičky. Funkcie na to určené vložia do paketu čísla portov programov *proxyse* a *proxycl*. Funkcia *odosli()* sa tiež stará o priebežné zvyšovania čísla udávajúceho poradie odoslaných paketov. *Odosli()* tiež obsahuje mechanizmus na odoslanie viac paketov naraz a následne čaká potvrdzujúce správy o prijatí. Maximálny počet nepotvrdených správ odoslaných naraz je možný nastaviť v súbore *konstanty.h*. Po pridaní interných hlavičiek sú dáta odoslané štandardnou funkciou *sendto()*.

Funkcia *prijmi()* sa stará o prijatie dát. Dáta sú prijímané štandardnou funkciou *recvfrom()*. Následne sa extrahujú informácie z prijatého paketu. Pomocné funkcie sa starajú o zistenie, či prijatý paket prišiel v správnom poradí, prípadne či sa jedná o správneho adresáta. Funkcia *prijmi()* posiela po každom prijatom pakete potvrdzovaciu správu. Ak sú dáta prijaté správne, sú postupne vkladané do vectoru a ten je predaný do programu, ktorý funkciu *prijmi()* volal.

Programy *preposlicl* a *preposlise* slúžia na simuláciu VoIP hovoru a preposielanie dát v rámci aplikácie. Po spustení si programy medzi sebou vymieňajú správy SIP. O ich obsahovú stránku sa

stará funkcia *sipx()*, kde x označuje poradie správy. Následne sa medzi programami vymieňajú správy RTP. Vloženie RTP hlavičky a veľkosti paketu sa stará pomocná funkcia *addrtp()*. V programoch beží nekonečná smyčka *while()*. Na kontrolu či prišli nejaké dáta na poslanie alebo treba poslať náhodné dáta je použitá štandardná funkcia *select()*. Časovač použitý v tejto funkcii využíva generátor pseudonáhodných čísel. Štandardná funkcia *rand()* v tomto prípade slúži na rozdielnu dobu odosielaní paketov.

Poslednú časť aplikácie tvorí program *proxyse*. Slúži na komunikáciu aplikácie s internetom. Program po prijatí prvej správy vytvorí proces. Ten sa bude starať o jedno spojenie. *Proxyse* funguje obdobne ako program *proxycl*. Beží v nekonečnej smyčke *while()* a prijíma dáta z *preposlise* alebo z internetu. Materský proces *proxyse* priebežne kontroluje návratové hodnoty skončených procesov.

7 Možnosti detekcie implementovaného útoku

Táto kapitola pojednáva o vlastnostiach aplikácie proxyvoip, ktorej návrh a implementácia boli popísané v predošlých kapitolách. Kapitola sa zameriava hlavne na spôsob skrytia dát a prípadnej detekcie výskytu takýchto dát v prenose.

Aplikácia proxyvoip sa snaží čo najvierohodnejšie napodobiť VoIP hovor. Preto je dôležité zistiť vlastnosti dátového prenosu aplikácie proxyvoip a následne ich porovnať s vlastnosťami skutočného VoIP hovoru. Pomôcť môžu aj nejaké existujúce aplikácie. Predovšetkým také, ktoré dokážu zistiť aké protokoly sa vyskytujú v prenášaných dátach. A tiež aplikácie slúžiace na prácu s multimédiami.

Cieľom aplikácie je snaha čo najviac sa podobať skutočnému hovoru. Preto dáta, ktoré aplikácia prenáša, sú RTP pakety podľa špecifikácie RTP protokolu. SIP správy, ktoré sú poslané na začiatku a konci činnosti aplikácie, tiež spĺňanú špecifikácie SIP protokolu.

Rozdiel medzi skutočným a falošným VoIP hovorom je obsah prenesených dát. Preto by bolo možné obsah tohto dátového prenosu extrahovať. V ďalšom kroku by nastala kontrola či tieto dáta v kontexte multimediálnych dát dávajú význam. To znamená zistiť či dáta reprezentujú hlas alebo video. Pretože skutočný VoIP hovor obsahuje ľudský hlas. A video obsahuje často obraz užívateľa alebo napríklad plochu počítača. V prípade rekonštrukcie dát, ktoré boli zaslané aplikáciou proxyvoip, sa multimediálne dáta sémanticky budú líšiť od bežných multimediálnych dát. Zvuk sa bude javiť ako určitý druh šumu. Bude sa vyskytovať praskanie a šušťanie ale nie zreteľná reč jedného z účastníkov simulovaného hovoru. Zachytené dáta reprezentujúce video prenos budú mať formu vizuálneho šumu. Tento spôsob má ale nevýhody v tom, že sa musí spracovať a sledovať kompletné pakety. Nestačí sledovať iba hlavičky alebo metadáta prenosu. Ďalší problém je, že treba určitým spôsobom definovať čo môže byť hlas a obraz prenášaný v VoIP hovoroch. Pre človeka je to jednoduché. Ale program musí vedieť rozlíšiť reč a obraz od náhodného šumu.

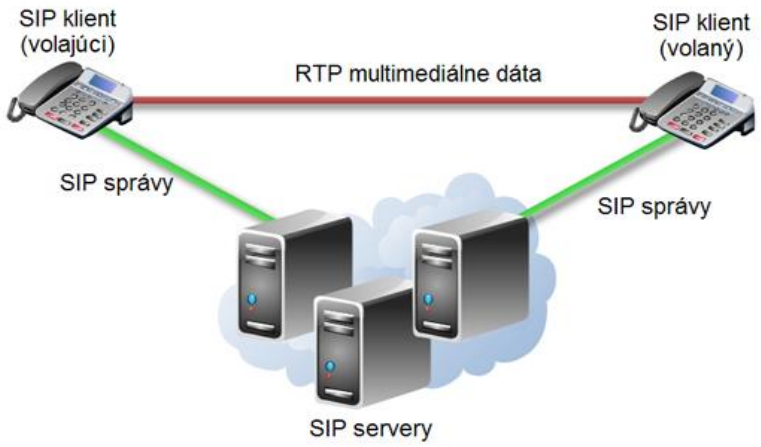
Na detekciu reči v zvukových dátach je možné použiť Voice Activity Detection algoritmus (VAD). Táto technika sa používa na detekciu rozprávania človeka, používa sa na zistenie či osoba hovorí v aktuálnom okamžiku alebo mlčí. Vtedy sa prenášajú len dáta označujúce ticho. A tým sa šetrí kapacita prenosu dát. VAD je možné použiť na zistenie či sa v zachytených dátach vyskytuje ľudská reč. Existuje šanca, že dáta prenášané aplikáciou budú dávať aj sémanticky význam ako zvukové – multimediálne dáta. Táto možnosť však je veľmi nepravdepodobná.

Iná možnosť detekcie skrytých dát je hľadanie výskytu špecifických dát v zachytenej komunikácii. Aplikácia proxyvoip používa v dátovom prenose interné hlavičky. Preto je možné použiť analýzu časti zachytených multimediálnych dát. A zistiť či by to nemohli byť iné dáta než

multimediálneho charakteru. Ak budú dávať význam dát internej hlavičky, tak sa môže jednať o aplikáciu proxyvoip a nie skutočný VoIP hovor. Problém je zistenie či zachytené dáta dávajú sémanticky význam interných hlavičiek. To znamená či zachytené čísla môžu označovať veľkosť dát alebo či to môže byť číslo portu soketu. Tomuto napríklad nevyhovejú záporné čísla a tiež čísla väčšie ako určitá medza. To či to môžu byť dáta reprezentujúce internú hlavičku je možné zistiť napríklad z podobnosti týchto dát vo viacero za sebou idúcich paketoch. Napríklad porty budú rovnaké pre pakety zasielané tesne po sebe. Prípadne čísla označujúce poradie paketov môžu túto detekciu tiež uľahčiť, pretože pakety musia byť postupne zoradené za sebou. Pomôcť môže tiež hľadanie pomocných správ aplikácie (potvrdzovacie správy). Podobne ako v predošlom spôsobe je veľmi nepravdepodobné, že skutočné multimediálne dáta by obsahovali dáta, ktoré sú sémanticky správne aj z hľadiska aplikácie proxyvoip. Významným nedostatkom tejto metódy je jej neobecnosť. Stačí ak sa interná hlavička zmení. Prípadne sa prehodí poradie polí internej hlavičky. Potom by sa musel rovnakým spôsobom upraviť aj spôsob detekcie.

Podobný spôsob tomu predošlému môže byť hľadanie dát popisujúcich pakety, ktoré aplikácia tuneluje. Jedná sa o hľadanie napríklad HTTP a HTTPS hlavičiek v zachytených dátach. Oproti predošlému je tento spôsob ešte menej vhodný. Pretože tieto dáta môžu ešte viac rozdelené a nachádzať sa na rôznych miestach v pakete. Prípadne môže byť medzi ne vložené interná hlavička a to sťažuje ich detekciu.

Spôsob detekcie skrytých dát v prenose môže spočívať aj v sledovaní určitých znakov, ktoré nie sú bežné VoIP hovory. Konkrétne sa jedná o fakt, že ústredňa a účastník majú iné IP adresy (viď schému na obrázku 18, na ktorom prebieha úvodná signalizácia s inými entitami než skutočný hovor). Aplikácia proxyvoip komunikuje iba medzi dvoma stranami a teda aj SIP a aj RTP správy sa prenášajú medzi tými istými IP adresami. Z uvedených informácií vyplýva, že takéto vlastnosti zachytených dát môžu značiť skryté dáta. Tento spôsob detekcie však môže byť považovaný ako pomocné rozhodovacie kritérium pre detekciu skrytých dát.



Obrázok 18 Schéma skutočného VoIP hovoru.

Ďalší spôsob odhalenia skrytých dát vo VoIP hovore je hľadanie výskytu RTCP [28] paketov. Tie slúžia na prenos štatistických a kontrolných informácií o prebiehajúcom VoIP hovore. Aplikácia proxyvoip však tieto pakety nepoužíva. Ale ani niektoré VoIP telefóny tieto pakety nepoužívajú. Preto sa môže jednať len o pomocné rozhodovacie kritérium.

Iný spôsob detekcie skrytých dát môže predstavovať analýza metadát prenosu [29]. Odposluch sa nezaujíma o samotné dáta, ktoré sa prenášajú vo vnútri paketov. Ale zameria sa na informácie ako sú veľkosť paketov, časy odoslania, IP adresy, čísla portov atď. Tieto informácie sa najskôr získajú zo skutočných VoIP hovorov. Na ich základe sa následne vytvorí profil VoIP hovoru. Jeho vlastnosti budú popísané štatistickými hodnotami metadát. Skúmaný VoIP hovor bude porovnaný s týmto profilom. Zistí sa ako veľmi sú si tieto veličiny podobné. V prípade odlišnosti sa môže jednať o prenos dát, ktorý sa iba chce podobať na VoIP hovor. Pre tento spôsob detekcie je dôležité nájsť dostatok správnych dát na vytvorenie správneho profilu. Až následne môže prebiehať rozhodovanie o pravosti kontrolovaných dát.

Toto boli uvedené niektoré spôsoby, ktoré by mohli detekovať skrytie dát. Ako najvhodnejšie sa javí kombinácia viacerých spôsobov. V nasledujúcej časti kapitoly bude uvedené, ktoré metódy budú implementované do aplikácie na detekciu skrytých dát v prenose.

7.1 Nástroj pre odhalenie ukrytých dát

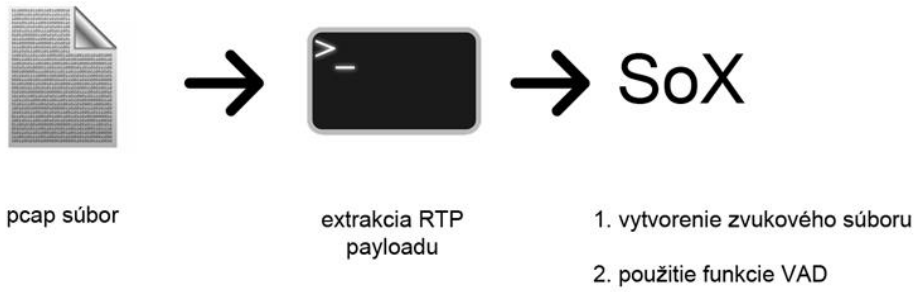
Na základe spôsobov detekcie popísaných v úvode kapitoly je navrhnutá aplikácia zistivoip. Bude slúžiť na zistenie či sa jedná o skutočný alebo falošný prenos dát vo forme VoIP hovoru.

Zistivoip bude na detekciu VoIP hovoru používať dva spôsoby. Prvý sa zameria na to, či SIP a RTP správy sa zasielajú medzi tými istými adresami. Toto je pomocné rozhodovacie kritérium.

Hlavný spôsob detekcie je zameraný na správnosť multimediálnych dát. Aplikácia zistivoip sa pokúsi zistiť či multimediálne dáta prenášané v paketoch RTP naozaj obsahujú ľudskú reč. Vstup aplikácie zistivoip tvorí pcap [30] súbor obsahujúci zachytenú sieťovú komunikáciu. Výstupom aplikácie je textový výpis oznamujúci či sa jedná o skutočný alebo falošný VoIP hovor. Falošný VoIP hovor slúži na prenos dát aplikácie proxyvoip. Obsahuje dáta protokolov HTTP a HTTPS namiesto ľudskej reči. Aplikácia zistivoip využíva pri svojej činnosti ďalší program, ktorý nie je dielom tejto diplomovej práce a je vytvorený inými autormi. Je to aplikácia SoX [1] a slúži na prácu s audio súbormi.

Detekcia bude prebiehať v niekoľkých krokoch (viď obrázok 19). Na začiatku aplikácia zistivoip načíta súbor vo formáte pcap. Následne ho bude prechádzať po jednotlivých paketoch. V prípade zistenia výskytu paketov protokolu SIP si uloží IP adresy zdroja a cieľu.

[1] SoX - Sound eXchange. *SoX - Sound eXchange* [online]. 2013 [cit. 2013-04-15]. Dostupné na URL http://sox.sourceforge.net/



Obrázok 19 Kroky aplikácie zistivoip

Ak sa nájdu pakety protokolu RTP, tak sa payload uloží do externého súboru. Tiež sa porovnajú adresy zdroja a cieľu protokolu RTP s uloženými adresami protokolu SIP. V prípade rovnakých IP adries SIP a RTP sa vypíše hlásenie, že nastala zhoda. Tým bolo splnená prvé rozhodovacie kritérium.

Pre druhé rozhodovacie kritérium aplikácia sleduje či RTP obsahuje skutočné audio dáta. Po extrakcii všetkých audio dát z RTP paketov sa program skončí. Nastáva fáza použitia aplikácie externej aplikácie. SoX skonvertuje súbor s vyextrahovanými dátami do zvukového súboru (k dátam pridá hlavičku a dáta následne prekóduje). Takýto súbor už je prehrateľný bežným softwarom na prehrávanie multimediálnych dát. Pomocou softwaru SoX sa následne zistí dĺžka ľudskej reči v audio súbore. Táto detekcia prebieha pomocou funkcie Voice Activity Detection (VAD). SoX odstráni z audio súboru úsek, na ktorom neprebieha hlasová aktivita. Následne sa zistí, či je dĺžka súboru väčšia ako nula. Ak je to nula, tak žiaden hlas prenášaný nebol a je pravdepodobné, že sa jedná o dáta programu proxyvoip. Ak je dĺžka väčšia ako nula je pravdepodobné, že sa jedná o skutočný VoIP hovor.

8 Testovanie

Overenie správnej funkčnosti aplikácie proxyvoip znamená overiť schopnosť tunelovania HTTP a HTTPS protokolov. A následne zistiť či sa dátový prenos aplikácie proxyvoip podobá skutočnému VoIP hovoru.

8.1 Tunelovanie protokolov

Tunelovanie protokolov HTTP a HTTPS je možné overiť komunikáciou medzi aplikáciou proxyvoip a internetovým serverom. Internetové stránky získané prostredníctvom aplikácie proxyvoip sú zobrazené rovnako správne ako keby boli získané priamym spojením bez použitia proxyvoip. Ale samotné získanie dát trvá o niečo dlhšie.

Veľkosť zdržania závisí na kvalite a rýchlosti spojenia jednotlivých úsekov komunikácie:

- Užívateľ – klientska časť aplikácie (v prípade, že sa nenachádzajú na tom istom počítači)
- Klientska časť – serverová časť
- Serverová časť – internetový server

Tabuľka 1 uvádza namerané hodnoty času (v sekundách) pri získavaní požadovaných dát. Priame spojenie značí prenos dát medzi užívateľom (prístup na internet prostredníctvom komerčného ISP na Slovensku) a internetovým serverom bez použitia aplikácie proxyvoip. Proxyvoip 1 a 2 značí komunikáciu s použitím aplikácie proxyvoip.

Tabuľka 1 Čas prenosu dát

	Protokol HTTP: načítanie stránky www.fit.vutbr.cz	Protokol HTTPS načítanie stránky www.vutbr.cz	Stiahnutie pdf súboru (2.1 MB protokol HTTP)
Priame spojenie	2,65	6,18	5,36
Proxyvoip 1	7,72	14,16	19,65
Proxyvoip 2	2,95	7,05	5,67

Legenda:
Priame spojenie - bez použitia aplikácie proxyvoip
Proxyvoip 1 - serverová časť aplikácie sa nachádza na serveri merlin.fit.vutbr.cz
Proxyvoip 2 - serverová časť aplikácie sa nachádza na lokálnej sieti

Z meraní vyplýva, že čas celkovej komunikácie najviac ovplyvňuje komunikácia medzi klientskou a serverovou časťou aplikácie proxyvoip. Rýchlosť prenosu dát medzi klientskou a serverovou časťou môže byť menšia ako umožňuje linka, pretože v skutočný VoIP hovor má relatívne nízku rýchlosť prenosu dát (ktorá závisí od použitých kodekov – napr. G.711: 64 kbit/s). Veľká prenosová rýchlosť by mohla upozorňovať na to, že sa nejdená o bežný VoIP hovor. Vyšší čas prenosu spôsobuje použitie interných hlavičiek a interného protokolu aplikácie proxyvoip.

8.2 Vlastnosti dátového prenosu

Nasledujúce obrázky zobrazujú porovnanie vlastností prenosu dát skutočného VoIP hovoru (použitie softvérových VoIP telefónov a ústredne Asterisk) a dátového prenosu aplikácie proxyvoip.

VoIP hovor

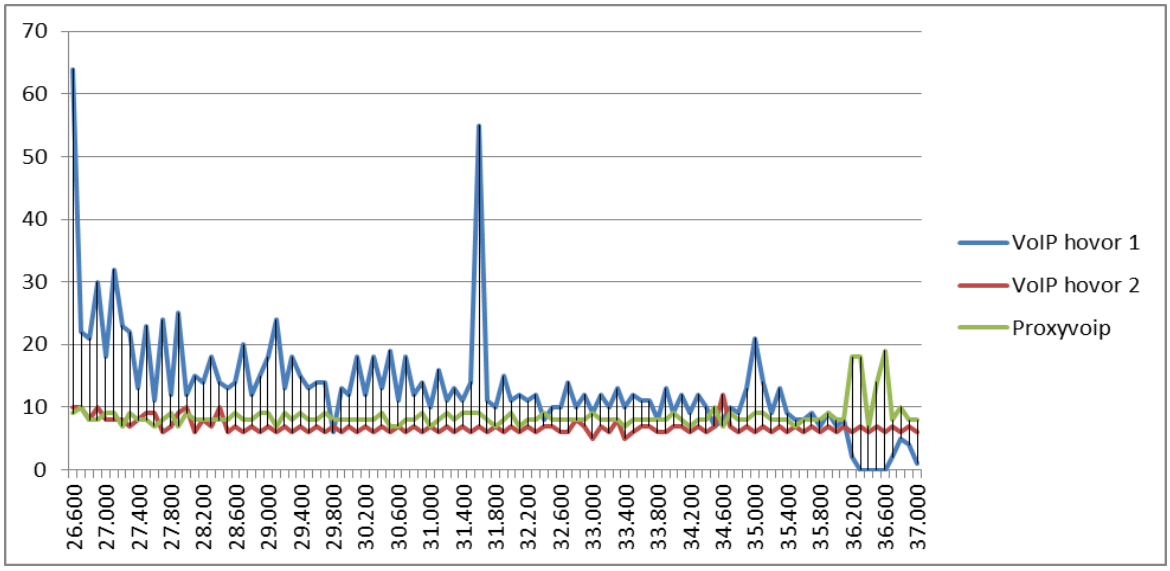
Topic / Item	Count	Rate (ms)	Percent
Packet Lengths	3721	0,084633	
0-19	0	0,000000	0,00%
20-39	0	0,000000	0,00%
40-79	1	0,000023	0,03%
80-159	1	0,000023	0,03%
160-319	2433	0,055338	65,39%
320-639	98	0,002229	2,63%
640-1279	845	0,019219	22,71%
1280-2559	343	0,007801	9,22%
2560-5119	0	0,000000	0,00%
5120-	0	0,000000	0,00%

Proxyvoip

Topic / Item	Count	Rate (ms)	Percent
Packet Lengths	5726	0,117472	
0-19	0	0,000000	0,00%
20-39	0	0,000000	0,00%
40-79	0	0,000000	0,00%
80-159	0	0,000000	0,00%
160-319	3704	0,075990	64,69%
320-639	0	0,000000	0,00%
640-1279	1069	0,021931	18,67%
1280-2559	953	0,019551	16,64%
2560-5119	0	0,000000	0,00%
5120-	0	0,000000	0,00%

Obrázok 20 Veľkosti paketov (v bajtoch)

Obrázok 20 ukazuje prenos dát skutočného VoIP hovoru a aplikácie proxyvoip z hľadiska veľkosti paketov. Najčastejšie sa prenášajú pakety o veľkosti 160-319 B a 640-1279 B, prípadne 1280-2559 B. Pakety s nižšou veľkosťou (160-319 B) predstavujú prenos audia a pakety s väčšou veľkosťou predstavujú prenos videa.



Obrázok 21 Počet prenesených paketov

Na obrázku 21 sú zobrazené počty prenesených paketov za jednotky času. Modrou a červenou farbou sú znázornené skutočné VoIP hovory. VoIP hovor 1 (modrá farba) dosahuje väčšie hodnoty, pretože obsahuje prenos viac video dát ako VoIP hovor 2. Dátový prenos aplikácie proxyvoip sa snaží napodobiť hodnoty počtu prenesených dát skutočných VoIP hovorov.

9 Zhodnotenie vytvorených nástrojov

Táto kapitola obsahuje zhodnotenie obidvoch vytvorených aplikácií. Prvá časť kapitoly pojednáva o aplikácii proxyvoip, ktorej úlohou je ukrytie dát. Druhá časť hodnotí aplikáciu zistivoip, slúžiacu na detekciu skrytia dát.

9.1 Nástroj na skrytie dát

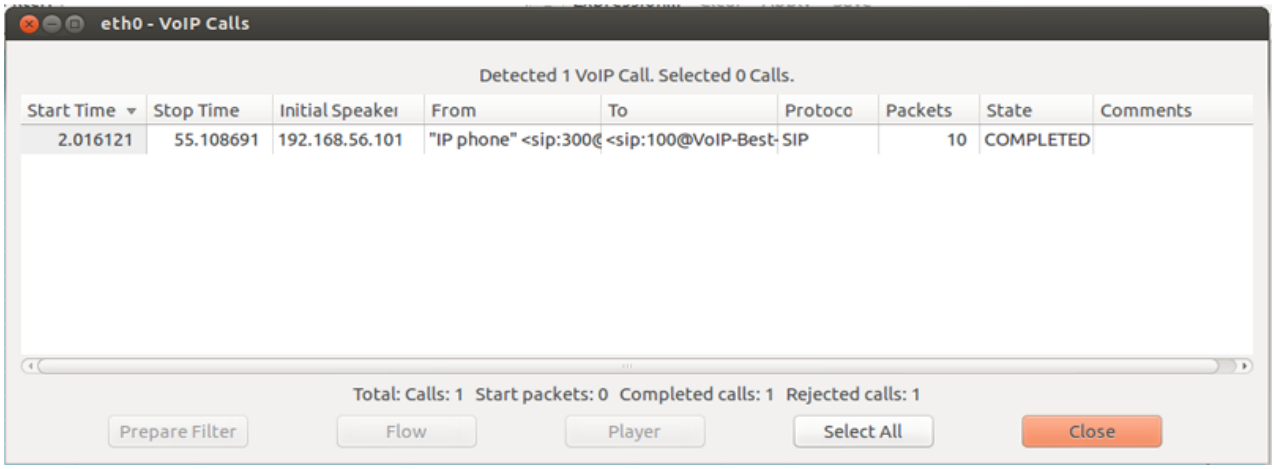
Aplikácia proxivoip slúži na maskovanie protokolov HTTP a HTTPS za VoIP hovor. Na zhodnotenie správnej funkčnosti aplikácie treba uvážiť dve kritéria. Prvé kritérium popisuje ako úspešne fungujú pôvodné protokoly. Následne bude popísaná úspešnosť s akou aplikácia maskuje, že sa jedná o multimediálne dáta VoIP hovoru.

Prvá časť hodnotenia sa zameriava na funkčnosť prehliadania internetových stránok tunelovaných cez proxyvoip. Táto vlastnosť sa dá overiť relatívne jednoducho. Stačí zadať do internetového prehliadača požadovanú adresu stránky a následne zistiť či sa stránka zobrazí korektne. Za základe používania aplikácie je možné povedať, že zobrazovanie internetových stránok funguje spoľahlivo. Správne funguje zobrazenie internetových stránok cez protokol HTTP a aj cez HTTPS (správne zobrazenie internetovej stránky http://www.fit.vutbr.cz/ je ukázané na obrázku 22). Jediný rozdiel z pohľadu užívateľa je rýchlosť zobrazenia požadovaných dát. To je spôsobené jednak rozdeľovaním a spájaním pôvodných dát ako aj pridanou internou komunikáciou (interné hlavičky a potvrdzovacie správy). Dôležité je aby toto časové zdržanie nebolo veľké. Tu je potrebné zdôrazniť, že veľkosť zdržania závisí od miesta kde sa serverová a klientska časť nachádzajú. Dôležité je ako dlho trvá prenos dát medzi týmito dvoma časťami aplikácie. V prípade, že je spojenie medzi nimi dostatočne kvalitné (rýchlosť prenosu rádovo v desiatkach megabitov a latencia rádovo v desiatkach mikrosekúnd), samotné zdržanie prenosu dát oproti priamej komunikácie (to znamená bez použitia aplikácie) je relatívne malé (viď merania v podkapitole 8.1).

Obrázok 22 Ukážka zobrazenia internetovej stránky

Extrémny prípad môže byť, že jednotlivé časti aplikácie sú spojené nekvalitným pomalým pripojením. Vtedy bude zdržanie badateľné. Ale v bežných podmienkach je zdržanie pri normálnych činnostiach nezávadné. Viac sa zdržanie môže prejaviť pri prenose väčšieho počtu dát.

Druhá časť hodnotenia spočíva v úspešnosti samotného maskovania dát za VoIP hovor. Pakety protokolov SIP a RTP sú korektné z pohľadu ich špecifikácie v príslušných RFC dokumentoch. Ďalšia možnosť je overenie fungovania aplikácie z pohľadu nejakej tretej strany. Na overenie úspešnosti simulovania RTP protokolu je možné použiť niektorý software, ktorý dokáže identifikovať VoIP hovor v zachytených dátach. Ako jeden z týchto nástrojov je možné uviesť *Wireshark*[2]. Je to software slúžiaci na vytváranie a prácu so zachytenou sieťovou komunikáciou. Pomocou rôznych pridaných funkcií dokáže detekovať o aký protokol sa jedná. Medzi tieto funkcie patrí aj schopnosť detekovať VoIP hovor. Práve funkcia detekcie VoIP hovorov identifikuje dátový prenos aplikácie ako VoIP hovor. Na obrázku 23 sa nachádza screenshot z analýzy komunikácie, ktorá prebiehala medzi klientskou a serverovou časťou aplikácie. Grafické porovnanie vlastností skutočného a falošného VoIP hovoru sa nachádza v podkapitole 8.2.



Obrázok 23 Ukážka detekcie aplikácie pomocou softwaru Wireshark

Ako ďalší program slúžiaci na detekciu VoIP hovoru je možné použiť program *VoIPong*[3]. Zameranie tejto aplikácie je detekcia a práca so zachytenou VoIP komunikáciou. Táto aplikácia nie je tak komplexná ako predtým uvedený *Wireshark*. Aj tento software detekuje dátový prenos aplikácie proxyvoip ako VoIP hovor (zobrazené na obrázku 24).

[2] Wireshark. Go deep. *Wireshark* [online]. 2013 [cit. 2013-04-17]. Dostupné na URL http://www.wireshark.org/

[3] *VoIPong* [online]. 2005 [cit. 2013-04-17]. Dostupné na URL http://www.enderunix.org/voipong/

```
Connected to VoIPong Management Console

System:
Ubuntu [Linux 3.2.0-40-generic #64-Ubuntu SMP Mon Mar 25
21:22:26 UTC 2013 i686]

voipong> shcall

ID   NODE1           PORT1 NODE2           PORT2 STIME   DURATION
.... ............... ..... ............... .... ......... .......
0305 147.229.212.12 8002  147.229.176.19 8000 18:11:38 65 sec

Total listed: 1
voipong>
```

Obrázok 24 Ukážka detekcie aplikácie pomocou softwaru VoIPong

Na základe týchto aplikácií je možné povedať, že vyvinutá aplikácia spĺňa schopnosť skryť dáta v sieťovej komunikácii.

Hodnotenie aplikácie a analýza jej vlastností ukazuje aj možnosti jej rozšírenia. Môže sa jednať o pridanie novej funkcionality alebo vylepšenie už nejakej existujúcej. Ako prvé rozšírenie sa javí možnosť doplniť aplikáciu o zasielanie RTCP správ. Tie môžu ešte viac pridať na vierohodnosti maskovania dát za VoIP hovor.

Ďalšie rozšírenie môže byť v optimalizácia interného protokolu. Dôkladnejším zamyslením sa nad ním je možné navrhnúť určité zlepšenia. Či už sa jedná o veľkosť jednotlivých polí hlavičiek, prípadne ktoré polia sa musia nutne nachádzať v týchto hlavičkách. Tiež je možné sa pokúsiť o optimalizáciu samotného spôsobu zasielania dát a ich potvrdenia o prijatí. Ako vhodná možnosť môže byť pridanie priebežného zasielania dát udávajúcich, že aktuálne spojenie je stále funkčné.

Následne je možné zlepšiť vierohodnosť RTP protokolu. Môže sa jednať o simulovanie rôznych multimediálnych dát. Prípadne sa môžu zlepšiť veľkosti a časy odoslania paketov, aby sa ešte viac približovali skutočným RTP paketom.

Napriek uvedeným možnostiam rozšírení a ideám na zlepšenie aplikácia spĺňa svoju funkčnosť tunelovania HTTP a HTTPS protokolov. A zároveň aj úspešne maskuje prenesené dáta za VoIP hovor.

9.2 Nástroj na detekciu skrytia dát

Aplikácia zistivoip pracuje so súbormi pcap, ktoré obsahujú zachytenú komunikáciu. Prvý krok je spoznanie RTP paketov medzi dátami. Následne prebieha extrakcia dát z paketov. V prípade, že sa nejaké dáta získajú, je možné zisťovať či sa jedná o skutočné multimediálne dáta.

Samotná extrakcia dát prebieha spoľahlivo. Dáta sa získajú podľa toho či majú RTP hlavičku. Ich skonvertovanie do audio súborov prebieha automaticky. Najdôležitejší krok je detekcia samotného zvuku. Použitá funkcia VAD sa snaží odlíšiť hlas od šumu. Toto môže byť niekedy menej presné. Ale na základe využitia viacerých hodnôt nastavenia funkcie VAD, detekcia hlasu pre potreby aplikácia zistivoip funguje správne.

```
**************************************************
Spusta sa analyza skutocneho VoIP hovoru.
**************************************************
Nic sa nenaslo.
Subor rtp0.dat obsahuje data s ssrc 9f602400
Subor rtp1.dat obsahuje data s ssrc 37752300
**Subor rtp0.dat bude spracovany.
Dlzka suboru:
9.860000
Dlzka hlasovej aktivity v subore:
6.360000
V subore sa vyskytuje hlasova aktivita.
**Subor rtp1.dat bude spracovany.
Dlzka suboru:
0.220000
Dlzka hlasovej aktivity v subore:
0.000000
V subore sa nevyskytuje hlasova aktivita.

Bola najdena aspon jedna hlasova aktivita v zdrojovych
suboroch. Preto sa pravdepodobne nejedna o skryte data.
```

Obrázok 25 Ukážka ako aplikácia pracuje s pcap súborom obsahujúcim skutočný VoIP hovor

Na obrázku 25 je vidieť ako zistivoip spracuje súbor so skutočnou VoIP komunikáciou. Úspešne extrahuje dáta a ich analýza ukáže, že sa jedná o skutočný VoIP hovor.

```
************************************************
Spusta sa analyza falosneho VoIP hovoru.
************************************************
Tie iste adresy su v sipe aj v rtp.
Subor rtp0.dat obsahuje data s ssrc 5493451e
Subor rtp1.dat obsahuje data s ssrc 37dea012
**Subor rtp0.dat bude spracovany.
Dlzka suboru:
9.240000
Dlzka hlasovej aktivity v subore:
0.000000
V subore sa nevyskytuje hlasova aktivita.
**Subor rtp1.dat bude spracovany.
Dlzka suboru:
11.300000
Dlzka hlasovej aktivity v subore:
0.000000
V subore sa nevyskytuje hlasova aktivita.

Nebola najdena ziadna hlasova aktivita a preto sa moze jednat
o prenos dat.
```

Obrázok 26 Ukážka ako aplikácia pracuje s pcap súborom obsahujúcim dáta, ktoré sa maskujú za VoIP hovor

Na obrázku 26 sa nachádza analýza dát získaných z komunikácie klientskej a serverovej časti aplikácie proxyvoip. Tieto dáta sú správne detekované ako nehlasové.

Dôležitý krok analýzy dát je vykonávaný externým softvérom. Tento program je však opensource a preto je možné nahliadnuť do jeho zdrojových súborov a zistiť ako pracuje. Pretože aj keď VAD funkcia pracuje správne, nedá sa vylúčiť, že niekedy nastane nesprávny výsledok. Skutočný hlas môže byť v zhoršených podmienkach nerozpoznaný a následne mylne vyhodnotený ako dáta aplikácie proxyvoip. Tak isto je možné, že niektoré dáta vytvoria náhodne zvuk, ktorý bude mylne detekovaný ako hlas.

Preto je možné v budúcnosti dôkladnejšie analyzovať túto funkciu VAD. Pokúsiť sa o jej zlepšenie. Takisto je možné skúsiť nájsť podobný algoritmus pre detekciu videa.

Budúce rozšírenie môže tiež spočívať v rozšírení práce s kodekmi. Zistivoip momentálne pracuje iba s kodekom G.711, pretože je to jeden z najčastejšie používaných VoIP kodekov a aj proxyvoip maskuje dáta práve za tento kodek. V budúcnosti by sa mohol z RTP paketov získať typ kodeku a pomocou SoX by sa z týchto dát pripravil univerzálny audio súbor (napríklad wav). Ten by slúžil na analýzu.

Prípadné ďalšie rozšírenie by mohlo byť doplnené o sledovanie metadát. Zo skutočných VoIP hovorov by sa vytvoril určitý profil VoIP hovoru a získané dáta by sa s ním porovnávali.

Ako ďalšie rozšírenie môže byť analýza dát práve prebiehajúcej komunikácie. Pri detekcii SIP a RTP paketov by sa extrahovalo napríklad prvých 10 sekúnd komunikácie a na nich by sa vykonala analýza.

Záver

Práca sa začína popisom systému zákonného odposluchu. Systém vychádza z noriem ETSI. Ďalej je uvedené z ktorých častí sa systém skladá a akým spôsobom sú dáta získané. Môže sa jednať iba o metainformácie alebo úplnú komunikáciu. Z pohľadu tejto diplomovej práce je najdôležitejší fakt, z ktorého miesta na sieti sú dáta získané. Tie sú poskytnuté od poskytovateľa pripojenia k sieti, cez ktorého odpočúvaný objekt komunikuje. Z toho vyplýva, že nie je spôsob ako sa odposluchu vyhnúť. Teoretická možnosť je použiť plne dedikovanú linku, ku ktorej majú prístup iba účastníci komunikácie. Toto však nie je bežný prípad sieťovej komunikácie.

Na základe informácií z noriem ETSI sú ďalej v práci ukázané techniky na skrytie dát v počítačovej sieti. Dáta sú skryté v iných dátach, tak aby neboli viditeľné pre nezúčastnenú osobu. A to aj v prípade, že odposluch bude mať úplnú komunikáciu k dispozícii. V práci je uvedených viacero spôsobov ako to dosiahnuť. Niektoré sú v praxi menej využiteľné, ale slúžia ako príklad.

Práca pokračuje popisom jedného zvoleného spôsobu skrytia dát v počítačových sieťach. Skrytie dát spočíva v maskovaní dát protokolov HTTP a HTTPS do falošného VoIP hovoru. Realizáciou vybraného spôsobu skrytia dát je aplikácia proxyvoip. Aplikácia je sprostredkovateľom komunikácie medzi užívateľom a internetovým serverom. Proxyvoip sa skladá z dvoch častí: klientskej a serverovej. Medzi obidvoma časťami prebieha falošný VoIP hovor. Správy protokolu SIP slúžia ako signalizácia začatia a ukončenia hovoru. Samotné dáta medzi klientskou a serverovou časťou sú prenášané v RTP paketoch. Ak užívateľ alebo internetový server nepožaduje prenos žiadnych dát, sú medzi klientskou a serverovou časťou sú prenášané náhodné dáta.

Dátový prenos medzi klientskou a serverovou časťou je detekovaný ako VoIP hovor dvoma externými aplikáciami Wireshark a VoIPong.

Použitie aplikácie proxyvoip spôsobuje menšie zdržanie komunikácie medzi užívateľom a serverom. Veľkosť zdržania závisí predovšetkým na kvalite a rýchlosti spojenia medzi klientskou a serverovou časťou aplikácie proxyvoip. Ale aj kvalita a rýchlosť spojenia medzi užívateľom a proxyvoip a medzi proxyvoip a internetovým serverom ovplyvňuje celkové zdržanie.

Práca následne popisuje spôsoby detekcie skrytých dát vo VoIP hovore. Najvhodnejší spôsob detekcie skrytých dát je zistenie, či sa v multimediálnych dátach VoIP hovoru nachádza ľudský hlas. Na základe tohto spôsobu bola vytvorená aplikácia zistivoip. Na detekciu ľudského hlasu sa využíva funkcia Voice Activity Detection externej aplikácie SoX.

Literatúra

[1] Tor Project: Anonymity Online. [online]. [cit. 2013-01-04]. Dostupné na URL https://www.torproject.org/

[2] ETSI TR 101 943. *Telecommunications security; Lawful Interception (LI); Concepts of Interception in a Generic Network Architecture.* Sophia-Antipolis Cedex, 2001.

[3] ETSI TR 101 944. *Telecommunications security; Lawful Interception (LI); Issues on IP Interception. Sophia-Antipolis Cedex*, 2001.

[4] WARF, Barney. Geographies of global Internet censorship. GeoJournal. roč. 76, č. 1, s. 1-23. ISSN 0343-2521. DOI: 10.1007/s10708-010-9393-3. Dostupné na URL http://link.springer.com/10.1007/s10708-010-9393-3

[5] SIMMONS, Gustavus. The History of Subliminal Channels. In: ANDERSON, Ross. *Information hiding: first international workshop, Cambridge, U.K., May 30-June 1, 1996: proceedings*. New York: Springer, c1996, s. 237-256. ISBN 3-540-61996-8.

[6] ZANDER, S., G. ARMITAGE a P. BRANCH. A survey of covert channels and countermeasures in computer network protocols. *IEEE Communications Surveys*. 2007, roč. 9, č. 3, s. 44-57. ISSN 1553-877x.

[7] CRAVER, Scott. On Public-Key Steganography in the Presence of an Active Warden. *Information Hiding*. 1996, s. 355-368. ISSN 0302-9743.

[8] LUCENA, Norka et al. Syntax and semantics-preserving application-layer protocol steganography. In: FRIDRICH, Jessica. *Information hiding: 6th international workshop*, IH 2004, Toronto, Canada, May 23-25 2004, revised selected papers. Berlin: Springer, 2004, s. 164-179. ISBN 978-3-540-24207-9.

[9] ISO/IEC 7498-1:1994(E). *Information technology - Open Systems Interconnection - Basic Reference Model: The Basic Model*. Switzerland: ISO/IEC Copyright Office, 1994.

[10] HANDEL, Theodore a Maxwell SANDFORD. Hiding Data in the OSI Network Model. In: ANDERSON, Ross. *Information hiding: first international workshop*, Cambridge, U.K., May 30-June 1, 1996 : proceedings. New York: Springer, c1996, s. 23-38.

[11] GRAF, Thomas. Messaging over IPv6 Destination Options. In: *Swiss Unix User Group* [online]. 2003 [cit. 2013-01-04]. Dostupné na URL http://grayworld.net/papers/messip6.txt

[12] WOLF, Manfred. Covert Channels in LAN Protocols. In: WORKSHOP LANSEC '89, Europ a T. Beth. *Local area network security: proceedings*. Berlin u.a: Springer, 1989, s. 91-101. ISBN 3-540-51754-5.

[13] GIRLING, C.G. Covert Channels in LAN's. *IEEE Transactions on Software Engineering*. 1987, SE-13, č. 2, s. 292-296. ISSN 0098-5589.

[14] FISK, Gina et al. Eliminating Steganography in Internet Traffic with Active Wardens. In: PETITCOLAS, Fabien A. *Information hiding: 5th international workshop, IH 2002, Noordwijkerhout, the Netherlands, October 7-9, 2002 : revised papers*. New York: Springer, c2003, s. 18-35. ISBN 3-540-00421-1.

[15] BUTTI, Laurent a Franck VEYSSET. Wi-Fi Advanced Stealth. In: *Black Hat US* [online]. 2006 [cit. 2013-01-04]. Dostupné na URL http://www.blackhat.com/presentations/bh-usa-06/BH-US-06-Veyssett.pdf

[16] KRÄTZER, Christian et al. WLAN steganography: a first practical review. In: *MM: proceedings of the Multimedia and Security Workshop 2006*, September 26-27, 2006, Geneva, Switzerland. New York, NY: ACM Press, c2006, s. 17-22. ISBN 1-59593-493-6.

[17] BOWYER, Lee. Firewall bypass via protocol steganography. [online]. 2002 [cit. 2013-01-04]. Dostupné na URL http://marc.info/?l=secpapers&m=103308908817830&w=2

[18] KUNDUR, Deepa a Kamran AHSAN. Practical Internet Steganography: Data Hiding in IP. In: *Proc. Texas Workshop on Security of Information Systems*. College Station, Texas, 2003.

[19] ZOU, Xin-guang et al. The research on information hiding based on command sequence of FTP protocol. In: KHOSLA, Rajiv, Robert J HOWLETT a L JAIN. *Knowledge-based intelligent information and engineering systems: 9th international conference, KES 2005, Melbourne, Australia, September 14-16, 2005: proceedings*. New York: Springer, c2005, s. 1079-1085. Lecture notes in computer science, 3681-3684. ISBN 978-3-540-28896-1.

[20] MAZURCZYK, Wojciech, Paweł SZAGA a Krzysztof SZCZYPIORSKI. Using transcoding for hidden communication in IP telephony. Multimedia Tools and Applications. ISSN 1380-7501.

[21] MOGHADDAM, Hooman et al. SkypeMorph: protocol obfuscation for Tor bridges. In: *Proceedings of the 2012 ACM conference on Computer and communications security*. New York, NY, USA: ACM, s. 97-108. ISBN 978-1-4503-1651-4.

[22] CRONIN, E., M. SHERR a M. BLAZE. On the (un)reliability of eavesdropping. *International Journal of Security and Networks*. 2008, roč. 3, č. 2, s. 103-113. ISSN 1747-8405.

[23] WARF, Barney. Geographies of global Internet censorship. In: GeoJournal. Volume 76, 2011, s. 1-23. ISBN 0343-2521ISSN 0343-2521.

[24] WINTER, Philipp a Stefan LINDSKOG. How the Great Firewall of China is blocking Tor. In: *Proceedings of the USENIX Workshop on Free and Open Communications on the Internet (FOCI 2012)*. 2012. Dostupné na URL https://www.usenix.org/system/files/conference/foci12/foci12-final2.pdf

[25] BENDRATH, R. a M. MUELLER. *The end of the net as we know it? Deep packet inspection and internet governance.* (August 4, 2010), 2010. Dostupné na URL http://www.cs.duke.edu/courses/common/compsci092/papers/govern/SSRN-id1653259.pdf

[26] RFC 3550. *RTP: A Transport Protocol for Real-Time Applications*. The Internet Society, 2003. Dostupné na URL http://www.ietf.org/rfc/rfc3550.txt

[27] RFC 3261. *SIP: Session Initiation Protocol*. The Internet Society, 2002. Dostupné na URL http://www.ietf.org/rfc/rfc3261.txt

[28] RFC 3605. *Real Time Control Protocol (RTCP) attribute in Session Description Protocol (SDP)*. The Internet Society, 2003. Dostupné na URL http://www.ietf.org/rfc/rfc3550.txt

[29] A Generic Technique for Voice over Internet Protocol (VoIP) Traffic Detection. In: IDREES, Fauzia a Uzma Aslam KHAN. *International Journal of Computer Science and Network Security*. 2008, s. 52-59.

[30] MARTINGARCIA, Luis. Tcpdump/Libpcap. *TCPDUMP/LIBPCAP public repository* [online]. 2010 [cit. 2013-04-15]. Dostupné na URL http://www.tcpdump.org/

Príloha A

Nastavenia a parametre spúšťania aplikácia proxyvoip:

Nastavenia

Pred prekladom aplikácie je možnosť zmeny niektorých parametrov. Sú uvedené v súbore *konstanty.h:*

- VELPAK – maximálna veľkosť paketu v bajtoch, ešte pred pridaním interných hlavičiek. Prednastavená hodnota je 1300.
- SEKUNDY a MIKROS – hodnoty času v sekundách a mikrosekundách slúžia na nastavenie čakacej doby na prijatie paketu v programoch preposli. Ak sa čas vyčerpá, pošlú sa náhodné dáta, aby simulovali VoIP hovor. Pôvodné hodnoty sú 0 a 15000.
- PRIJMI_SEK a PRIJMI_MIKROS – hodnoty sekúnd a mikrosekúnd časovača použitého v programoch proxy. Ak počas tejto doby druhá strana neodpovie, programy sa ukončia. Prednastavené hodnoty sú 10 a 30000.
- OPAK – slúži ako hodnota maximálne počtu chýb. Použitá je vo funkciách *odosli()* a *prijmi()*. Ak počet chýb (paket mimo poradia, vypršaný časový interval, nesprávny potvrdzovací paket) bude vyšší ako táto hodnota, prenos sa ukončí. V tom prípade sa vracia chybová návratová hodnota. Pôvodná hodnota je 3.
- OKNO – hodnota udávajúca maximálny počet odoslaných paketov bez potvrdzovacej správy. Využíva sa vo funkciách *odosli()* a *prijmi()*. Pôvodná hodnota je 5.
- VYPIS – zapína alebo vypína kontrolné výpisy programov. 0 znamená žiadne výpisy, 1 iba základné výpisy a 2 vypisuje každú zaznamenanú činnosť. Pri hodnote 0 sa vypisujú iba prípadné chybové hlášky. Predvolená hodnota je 0.

V prípade zhoršenej kvality spojenia je možné upraviť hodnoty nasledovných konštánt: PRIJMI_SEK a PRIJMI_MIKROS sa doporučuje zvýšiť o niekoľko sekúnd, OPAK zvýšiť napríklad na hodnotu 5 a OKNO znížiť až na hodnotu 1. Ak je hodnota OKNO nastavená na 1, po každom odoslanom pakete sa čaká na potvrdzovaciu správu. Z hľadiska funkčnosti aplikácie pri zhoršených podmienkach je to dobré nastavenie, ale môže to zvýšiť čas potrebný na prenos dát.

Spustenie aplikácie

Všetky programy vyžadujú spúšťanie s parametrami. Tie udávajú adresu alebo číslo portu. Ako prvé sa spúšťajú programy serverovej časti. Programy klientskej časti sa spúšťajú až následne, pretože sa pripájajú k bežiacim programom serverovej časti.

Serverová časť:

./preposlise *port_preposlicl adresa_proxyse port_proxyse1 port_proxyse2*

- *port_preposlicl* je číslo portu programu na komunikáciu s preposlicl, *adresa_proxyse* je internetová adresa a *port_proxyse1* je port, na ktorom sa nachádza proxyse, *port_proxyse2* je port programu slúžiaci na komunikáciu s proxyse

./proxyse *port*

- *port* udáva číslo portu programu, na ktorom prijíma spojenia od preposlise

Klientska časť:

./preposlicl *port_prepolsicl port_preposlise adresa_preposlise port_proxycl*

- *port_preposlicl* je port programu na komunikáciu s preposlise, *port_preposlise* a *adresa_preposlise* je port a internetová adresa, kde sa nachádza preposlise, *port_proxycl* je port programu slúžiaci na komunikáciu s proxycl

./proxycl *port_browser adresa_preposlicl port_preposlicl port_preposlicl2*

- *port_browser* udáva port programu, na ktorom čaká spojenia od internetového prehliadača, *adresa_preposlicl* a *port_preposlicl* je internetová adresa a port, na ktorom sa nachádza preposlicl, *port_preposlicl2* je port programu slúžiaci na komunikáciu s preposlicl

Príloha B

Obsah DVD

Zložky:

- technická správa – obsahuje text diplomovej práce vo formátoch docx a pdf
- ubuntu – obsahuje upravený image súbor s operačným systémom Ubuntu
- zdrojové súbory – obsahuje zdrojové súbory aplikácií proxyvoip a zistivoip

Súbor:

- readme.txt – návod na vyskúšanie aplikácii proxyvoip a zistivoip pomocou upraveného image súboru s operačným systémom Ubuntu